Methoden und Werkzeuge zur Verfügbarkeitsermittlung

Miroslaw Malek, Günther A. Hoffmann, Nikola Milanovic,
Stefan Brüning, Reinhard Meyer, Bratislav Milic
{malek, gunho, milanovi, bruening, rmeyer, milic} @informatik.hu-berlin.de

Humboldt-Universität zu Berlin
Institut für Informatik
Rechnerorganisation und Kommunikation

Inhaltsverzeichnis

1 Methoden zur Verfügbarkeitsermittlung

1.1 Definitionen

Im Bereich Hochverfügbarkeit existieren vielfältige Definitionen von Konzepten, Variablen und Parametern. Einerseits führen Autoren in ihren Publikationen eigene Definitionen ein, um Mehrdeutigkeiten zu entgegnen, andererseits werden auch umgangssprachliche Formulierungen übernommen, die Interpretationsspielräume zulassen. Des Weiteren werden in wissenschaftlichen Kreisen unterschiedliche Definitionen zum Beispiel für *Verfügbarkeit* eingesetzt oder es konnte sich noch keine Definition durchsetzen. Im folgenden Abschnitt stellen wir die, aus Sicht der durchzuführenden Studie, wesentlichen Definitionen in alphabetischer Reihenfolge zusammen. Die vorgestellten Definitionen orientieren sich an, im internationalen wissenschaftlichen Umfeld, etablierten Definitionen, die meist in Englisch vorliegen. Zur weiteren Einschränkung von Mehrdeutigkeiten stellen wir daher neben den deutschsprachigen auch die englischsprachigen Definitionen zur Verfügung.

1.1.1 Tabellarische Darstellung

Deutsch	***Englisch***
Verfügbarkeit, synonym auch availability *Verfügbarkeit* von Systemen kann auf mehrere Arten definiert werden, die im folgenden detailliert dargestellt werden. Verfügbarkeit ist eng mit *Zuverlässigkeit* verknüpft und wird in den Empfehlungen der ITU-T E.800, International Telecommunication Union, wie folgt definiert: Die Fähigkeit einer Einheit, eine angeforderte Funktion zu jedem beliebigen Zeitpunkt innerhalb eines vorgegebenen Zeitintervalls erfolgreich durchzuführen, unter der Voraussetzung, dass externe Ressourcen,	**Availability** There are a number of definitions for system *availability*. *Availability* is closely related to *reliability*, and is defined in ITU-T Recommendation E.800 as follows: The ability of an item to be in a state to perform a required function at a given instant of time or at any instant of time within a given time interval, assuming that the external resources, if required, are provided (ITU-T E.800). An important difference between *reliability* and

Deutsch	***Englisch***
soweit gefordert, vorhanden sind (ITU-T E.800). Zwischen *Zuverlässigkeit und Verfügbarkeit* besteht ein bedeutsamer Unterschied: *Zuverlässigkeit* ist die Wahrscheinlichkeit einer fehlerfreien Operation zu einem gegebenen Zeitpunkt *t,* dass das System zum Zeitpunkt t=0 fehlerfrei arbeitete. *Verfügbarkeit* dagegen bezieht sich auf einen bestimmten Zeitpunkt, typischerweise dem Zeitpunkt, an dem das System tatsächlich angesprochen wird, um eine bestimmte Funktion zu erfüllen. *Unmittelbare Verfügbarkeit* (*instantaneous availability*) A_i ist die Wahrscheinlichkeit, dass das System zum Zeitpunkt t korrekt funktioniert. Zu beachten ist, dass in Abwesenheit von Reparatur oder Ersatz, *unmittelbare Verfügbarkeit* mit der *Zuverlässigkeit R* gleichzusetzen ist. $A_i(t) = R(t)$ (1.1.1.1) Die S*teady-State-Verfügbarkeit* A_{ss} (unserer Einsicht nach *Stationäre Verfügbarkeit*, der Begriff wird im Deutschen jedoch kaum verwendet*)* ist die Wahrscheinlichkeit, dass das System zu jedem beliebigen Zeitpunkt funktionsfähig ist. Dies wird durch das Verhältnis von funktionaler Zeit (*uptime*) zur	*availability* is that reliability is the probability of failure-free operation at a given time t provided that the system was operational at time t=0, while availability refers to failure-free operation at a given instant of time, usually the time when a device or system is accessed to provide a required function or service. *Instantaneous availability* A_i is the probability that a system is performing correctly at time t. Note that in the absence of a repair or a replacement, it is equal to the *reliability R*. $A_i(t) = R(t)$ (1.1.1.6) The *steady-state availability* A_{ss} is the probability that a system will be operational at any random point in time and is expressed as the fraction of time a system is operational during its expected lifetime. $A_{ss}(t) = \frac{uptime}{lifetime}$ (1.1.1.7) *Interval call availability* A_c which is the probability that calls within a specific time interval Δt in a given system or component will be handled correctly.

Deutsch	***Englisch***
gesamten Lebenszeit (*lifetime*) des Systems ausgedrückt. $A_{ss}(t) = \frac{uptime}{lifetime}$ (1.1.1.2) *Interval-Call-Verfügbarkeit* A_c ist die Wahrscheinlichkeit, dass innerhalb eines spezifischen Zeitintervalls Δt eingehende Systemanrufe korrekt ausgeführt werden. $A_c(\Delta t) = 1 - \frac{n_{failed}}{n_{total}}$ (1.1.1.3) *Interval-Call-Verfügbarkeit* A_c berechnet sich aus der Zahl Eins abzüglich dem Quotienten aus fehlgeschlagenen Anrufen (n_{failed}) und der Gesamtzahl eingegangener Anrufe (n_{total}) innerhalb des Zeitintervalls Δt. Das entspricht dem Quotienten aus Anzahl der korrekt abgewickelten Anrufe ($n_{completed}$) und der Gesamtzahl eingegangener Anrufe (n_{total}) innerhalb von Δt. $A_c(\Delta t) = \frac{n_{completed}}{n_{total}}$ (1.1.1.4) In Anlehnung an den Begriff *Interval-Call-Verfügbarkeit* kann *Service-Verfügbarkeit* als Wahrscheinlichkeit definiert werden, dass	$A_c(\Delta t) = 1 - \frac{n_{failed}}{n_{total}}$ (1.1.1.8) It is calculated as one minus the number of failed calls n_{failed} over the total number of calls n_{total} within Δt . This is equivalent to the number of correctly completed calls $n_{completed}$ over the total number of calls n_{total} . $A_c(\Delta t) = \frac{n_{completed}}{n_{total}}$ (1.1.1.9) Similarly to Interval call availability Ac , service availability can be defined as the probability that service requests within a specific time interval Δt will be handled correctly. $A_S(\Delta t) = 1 - \frac{n_{failed}}{n_{total}}$ (1.1.1.10) where n_{total} is the total number of service requests within Δt and n_{failed} is the number of failed service request within Δt. *Service Availability* is the availability of a service as measured by the end user. It includes the availability of all hardware and software elements, computation, network and storage systems needed in order for the service to be

Deutsch	***Englisch***
innerhalb eines Zeitintervalls Δt eingehende Serviceanfragen korrekt bearbeitet werden. $A_S(\Delta t) = 1 - \frac{n_{failed}}{n_{total}}$ (1.1.1.5) Hierbei ist n_{total} die Gesamtzahl eingehender Serviceanfragen und n_{failed} die Anzahl der fehlgeschlagenen Serviceanfragen. *Service-Verfügbarkeit* ist die Gesamtheit sämtlicher Elemente, die an der Erbringung des Services involviert sind. Dies inkludiert sämtliche Hardware- und Software-Elemente, Recheneinheiten, Netzwerk, Speicher-komponenten, etc., die das System zur korrekten Leistungserbringung benötigt. Manche dieser Elemente können hochverfügbar, müssen es jedoch nicht sein. Beachten Sie den Unterschied zur *Verfügbarkeit*, die auf jede Komponente innerhalb des Gesamtsystems angewendet werden kann. Abzugrenzen ist Service-Verfügbarkeit vom Endanwendermonitoring (EAM), welches hauptsächlich Quality-of-Service-Maße (QoS) betrachtet.	delivered correctly. *Service availability* is the product of the availability of all elements, some of which may be highly available, involved in providing the service. This differs from general high *availability* which can be discussed on an element-by-element level.
Verlässlichkeit *Verlässlichkeit* wird als Überbegriff von Attributen wie *Zuverlässigkeit, Verfügbarkeit, Betriebssicherheit, Betriebsschutz, Integrität*	**Dependability** Defined as the trustworthiness of a computing system which allows reliance to be justifiably placed on the service it delivers. *Dependability*

Deutsch	***Englisch***
und weiteren eingesetzt (IFIP WG 10.4). Verlässlichkeit beschreibt die Qualität eines Services, den das betrachtete System bereithält. Die oben genannten Metriken quantifizieren diese Konzepte.	thus includes as special cases such attributes as *reliability*, *availability*, *safety*, *security* (IFIP WG 10.4) *Dependability* is a term used to encapsulate the concepts of *reliability, availability, integrity, maintainability, performability, safety and testability*. In other words, dependability is simply the quality of service provided by a particular system and terms such as *reliability, safety, availability, maintainability, performability*, and *testability* are examples of measures used to quantify it. The *dependability* of a system or service indicates how much reliance can justifiably be placed on it.
Fehlerfrüherkennung Fehlerfrüherkennung beschreibt die Fähigkeit, Fehler in einem System zu entdecken, bevor diese zu Ausfällen führen. Diese Definition steht im Gegensatz zur Definition von Holzmann (1996), der unter Fehlerfrüherkennung eine Technik beschreibt, die in der frühen Phase einer Softwareentwicklung den Entwickler auf mögliche Probleme und Gefahren im Programmentwurf hinweist.	**Early Fault Detection** By *Early Fault Detection* we mean the ability to detect faults in a system before these faults generate errors. This is in *contrast* to the definition by Holzmann (1996) who proposes the same term for a tool or technique that can be applied in the early phase of a still incomplete and imprecise software design, and that can provide the user with some guidance about potential hazards in the design process.
Fehler, Störung, Ausfall *Fehler* produzieren *Störungen*, die zu *Ausfällen* führen können. Ein *Ausfall* tritt dann auf, wenn	**Fault, Error, Failure** A *failure* occurs when the delivered service deviates from the specified service: failures are

Deutsch	***Englisch***
der gelieferte Service vom spezifizierten Service abweicht. Eine *Störung* ist die Manifestation eines Fehlers in einer Software, einem Modul oder einer Komponente (Siewiorek and Swarz (1992)).	caused by *errors*. A *fault* is an incorrect state of hardware or software. An *error* is a manifestation of a fault within a program or data structure forcing deviation from the expected result of computation (incorrect result). *Faults* are the cause of *errors* that may lead to *failures* (Siewiorek and Swarz (1992)).
Mean Time Between Failures (MTBF) Bei der durchschnittlichen Zeit zwischen Ausfällen wird von einem Erneuerungsprozess für wiederherstellbare Systeme ausgegangen. MTBF wird, wie folgt, berechnet: MTBF = MTTF + MTTR (1.1.1.11) Siehe hierzu die Definitionen von MTTF und MTTR. Bei der Verwendung von MTBF und MTTF ist Vorsicht geboten, beide Werte werden oft synonym verwendet. Dies gilt jedoch nur bei sehr kleinen MTTR Werten.	**Mean Time Between Failures (MTBF)** Mean (or average) Time Between Failures. MTBF assumes a renewal process for a repairable system and is estimated as the total number of equivalent (system or element) hours divided by the total number of failures. For a renewal process, at the time equal to the MTBF, the expected number of failures will be equal to the starting sample size. This does not mean that every system will have at least one failure. For a homogeneous Poisson process (HPP), the distribution of failures for a sample of systems can be calculated via the Poisson distribution model using the estimated MTBF. MTBF = MTTF + MTTR (1.1.1.12) Warning: MTBF and MTTF are often used interchangeably, and care should be taken, as they cease to be equivalent if MTTR becomes significant.

Deutsch	*Englisch*
Mean Time To Failure (MTTF) Für nicht-wiederherstellbare Systeme ist MTTF die durchschnittliche Zeit bis ein Ausfall auftritt. Der Term MTBF ist geläufiger, gilt jedoch nur dann, wenn MTTR-Werte sehr viel kleiner sind als MTTF-Werte. Siehe hierzu auch die Definition von MTBF.	**Mean Time To Failure (MTTF)** For non-repairable systems or elements, the MTTF is the average time to failure of the population distribution of lifetimes measured from time zero. For repairable systems, the MTTF is average time between sequential failures. For repairable systems, the term MTBF is more commonly used, especially when repair times are relatively small and can be neglected. Only for the exponential distribution (constant failure rate) does 1/MTTF (or 1/MTBF) equal the failure rate (IFR). In general, for any other failure distribution, the failure rate (IFR or hazard rate) must be determined using f(t)/[1-F(t)], where f(t) is the PDF and F(t) is the CDF. PDF is the probability density function, CDF is the conditional density function.
Mean Time To Repair (MTTR) Die durchschnittliche Zeit, um eine Komponente oder ein System wiederherzustellen, gemessen über eine ausreichend große Messmenge.	**Mean Time To Repair (MTTR)** The average time it takes to repair an element or system as measured (or projected) over a large number of repairs.
Modell Ein Modell beschreibt einen Prozess oder ein System auf einer kleineren, mit verfügbaren Methoden und Werkzeugen handhabbaren Detailstufe. Man unterscheidet drei prinzipielle Modellierungsverfahren anhand des	**Model** *Etymology*: from old Italian modello, from (assumed) vulgar Latin modellus, from Latin modulus small measure, middle French modelle (1) a simplified description of a complex entity or

Deutsch	***Englisch***
Detailwissens, das in die Modellbildung einfließt: **Black-Box**: Modellgenerierung ohne Kenntnis interner Systemdetails. Häufig basierend auf quantitativer Systembeobachtung und statistischer Analyse oder Verfahren des maschinellen Lernens. **Grey-Box**: Nutzt teilweise Wissen aus internen Funktionsabläufen sowie funktionalen Zusammenhängen und kombiniert dieses mit Messungen aus dem System (siehe Black-Box). **White-Box**: Kenntnisse über die internen Funktionsabläufe und funktionalen Zusammenhänge fließen in die Modellbildung ein. Dies kann beispielsweise Wissen über Architekturen und Konzepte, Konfigurationen oder organisatorische Maßnahmen sein.	process; "the computer program was based on a model of the circulatory and respiratory systems" (Princeton Dictionary) (2) the act of representing something (usually on a smaller scale) (Merriam-Webster). We add to this on „smaller detail“. **Black-Box**: measurement and statistical interference based on observation only (e.g. black box testing where test are applied and outputs are observed without knowledge of a given system). **Grey-Box**: Considers partial knowledge about architecture to derive a model of the system. **White-Box**: Considers knowledge about architecture to derive a model of the system
Leistungsfähigkeit Systemleistung unter Ausfall- und Reparaturszenarien.	**Performability** System performance under failure and repair
Precision Precision ist der Quotient aus korrekten Vorhersagen (wahr positiv – TP) und der Summe aus wahr positiven und falsch positiven Vorhersagen (TP+FP).	**Precision** Precision is the ratio between the number of correct predictions (*TP* – true positives), e.g. failure prediction, and the sum of true positive and false positive (*TP*+*FP*) predictions.

Deutsch	***Englisch***
$precision = \frac{TP}{TP + FP}$ (1.1.1.13)	$precision = \frac{TP}{TP + FP}$ (1.1.1.14)
Recall *Recall* ist der Quotient aus korrekten Vorhersagen (wahr positiv – TP) und der Summe aus wahr positiven und falsch negativen (*TP+FN*) Vorhersagen. *Recall* wird synonym auch *Sensitivität* oder *Positive Accuracy* genannt. $recall = \frac{TP}{TP + FN}$ (1.1.1.15)	**Recall** Recall is defined as the ratio between correct predictions (*TP*) and total number of incidents, e.g. failures, (*TP+FN*) that have occurred. Recall sometimes is also called *sensitivity* or *positive accuracy*. $recall = \frac{TP}{TP + FN}$ (1.1.1.16)
Zuverlässigkeit *Zuverlässigkeit* wird umgangssprachlich häufig als Überbegriff eingesetzt, stattdessen wird jedoch *Verlässlichkeit* zunehmend im akademischen und wirtschaftlichen Umfeld eingesetzt, da der Begriff *Zuverlässigkeit* bereits als mathematische Funktion definiert ist. *Zuverlässigkeit* ist definiert als die Wahrscheinlichkeit, dass ein Produkt, ein System oder eine Komponente die intendierte Funktion über einen spezifizierten Zeitraum und unter definierten Bedingungen ausführt. Die Zuverlässigkeitsfunktion *R*(t) ist definiert als die zeitabhängige Wahrscheinlichkeit, dass ein zufällig aus einer Menge gewähltes Produkt	**Reliability** *Reliability* is used colloquially as an umbrella term but more correctly it should be called *dependability*. *Reliability* is also used as a mathematical function. *Reliability* is an abstract term and is defined as the probability that a product (system) performs its intended function for a specified time period when operating under normal (or stated) environmental conditions. The reliability function, *R*(t), is defined as the time dependent conditional probability that a random product (or system) drawn from a population performs correctly throughout the interval of time $[t_0, t]$, given that the product (system) was performing correctly at time t_0.

Deutsch	***Englisch***
oder System über einen Zeitintervall $[t_0, t]$ seine Funktion korrekt ausführt, mit dem Wissen, dass das System zum Zeitpunkt t_0 korrekt arbeitete. *Zuverlässigkeit* unterscheidet sich von Verfügbarkeit: Bei *Zuverlässigkeit* betrachtet man genau ein Ereignis, und zwar den Ausfall (failure). Bei *Verfügbarkeit* betrachtet man zwei Ereignisse, den Ausfall und die Wiederherstellung. Ein System kann hochverfügbar sein und trotzdem häufig durch inoperable Perioden gehen, sofern diese nur kurzzeitig auftreten. Zur deutlichen Unterscheidung zwischen *Verlässlichkeit* und *Zuverlässigkeit* hat man sich bei der International Federation for Information Processing (IFIP WG 10.4) darauf verständigt, *Verlässlichkeit* als Überbegriff und *Zuverlässigkeit* als mathematische Funktion einzusetzen.	*Reliability* differs from *availability* in that reliability involves only one event, failure, while availability takes into account two events, failure and recovery. A system can be highly available yet experience frequent periods of inoperability as long as the length of each period is short. To remove confusion, IFIP WG 10.4 (IFIP WG 10.4) has proposed *dependability* as an umbrella term and *reliability* is to be used a mathematical function.
Service [1] Die Bezeichnung Service wird in unterschiedlichen Kontexten eingesetzt, nicht nur im IT-Bereich. Häufig wird ein Service als „eine Einrichtung, die einen Bedarf deckt“ definiert. Ein Service kann daher von Unternehmen, Einzelpersonen, einem System oder einer Komponente erbracht werden.	**Service** [1] Service is used in multiple contexts, not only in reference to IT systems. Merriam Webster OnLine defines a service as “a facility supplying some public demand”. This implies that a service can be performed by anyone (e.g. craftsman, enterprises as well as hardware systems or software) as long as we have the notion of a service consumer on one side and a

Deutsch	*Englisch*
Um diese breite Definition auf IT-Systeme einzuschränken, wird die zu erbringende Leistung häufig auf immaterielle Leistungen eingeschränkt. Dieser Definitionsansatz kann weitergeführt werden: Services werden von Computerprogrammen für andere Computerprogramme, Komponenten oder für den Endnutzer erbracht. Ein Service ist dann ein in sich geschlossenes Modul, welches entweder für andere maschinelle Einheiten oder für den Endnutzer eine Leistung erbringt. [2] Ein Service stellt eine Anzahl von Aktivitäten dar, um Anfragen maschineller Systeme oder eines Endnutzers zu erfüllen. Zum Beispiel bearbeiten Datenbankserver Aktualisierungs-, Einfügungs- und Löschanfragen; Sicherheitsserver bearbeiten Anfragen bezüglich Authorisation und Authentifikation.	service provider on the other side. One can limit this wide spectrum of service vendors adding the phrase: a service “does not produce a tangible commodity”; which leads more into the direction of IT systems. This focus can be strengthened by denoting a service as a “meaningful activity that a computer program performs at the request of another computer program [...] A service is thus a remotely accessible, self-contained application module”. A service can use other services to perform its task and it can also be addressed locally. That guides to a software-focused definition where “a service is a piece of software that computes a certain task on request of another piece of software or a human user” (Krafzig et al. 2004). [2] A service is a set of actions that satisfy a request from a user or other system. For example, database servers satisfy requests for updates, inserts or other database actions; security servers satisfy requests such as authorization and authentication.

1.1.2 Ordnungsprinzip

Die im vorherigen Kapitel definierten Begriffe und Konzepte werden im akademischen Umfeld in der in Abbildung 1 dargestellten Abhängigkeit zueinander diskutiert. Verlässlichkeit ist der Überbegriff, dem sich Attribute, Bedrohungen und die Art und Weise des Umgangs mit der Bedrohung unterordnen Avizienis et al. (2003).

<table>
<tr><td rowspan="11">Verlässlichkeit
Dependability</td><td rowspan="3">Attribute
Attributes</td><td rowspan="2">Sicherheit
Security</td><td>Verfügbarkeit
Availability</td></tr>
<tr><td>Vertraulichkeit
Confidentiality
Integrität
Integrity</td></tr>
<tr><td colspan="2">Zuverlässigkeit
Reliability
Wartbarkeit
Maintainability
Betriebssicherheit
Safety</td></tr>
<tr><td colspan="3"></td></tr>
<tr><td rowspan="3">Art und Weise
Means</td><td colspan="2">Fehlerprävention
Fault Prevention
Fehlerentfernung
Fault Removal
Fehlertoleranz
Fault Tolerance
Fehlervorhersage (überwiegend software)
Fault Forecasting</td></tr>
<tr><td colspan="3"></td></tr>
<tr><td rowspan="3">Bedrohungen
Threats</td><td colspan="2">Fehler
Faults
Störung
Errors
Ausfall
Failures</td></tr>
</table>

Abbildung 1: Abhängigkeit der Begriffe und Konzepte.

1.2 Auswahl und Bewertungskriterien der Methoden

Eine Vielzahl von Modellierungstechniken sind bereits auf Software- und Hardwaresysteme angewendet worden, unter anderem analytische Modelle sowie quantitative und qualitative Ansätze. Im folgenden Abschnitt werden wir uns auf Techniken konzentrieren, die bereits im industriellen Umfeld dokumentiert und eingesetzt wurden.

Zur Bewertung wurden folgende Kriterien eingesetzt:

1. Beschreibung der Methode

2. Eingabewerte der Methode
    - Wissen über funktionale Zusammenhänge (z.B. Black-Box, Grey-Box, White-Box, Verteilungen, Übergangswahrscheinlichkeiten)
    - Messungen (z.B. Zeitreihen, kategorische Daten, Übergangswahrscheinlichkeiten)
3. Ausgabe der Methode (z.B. Modell, Kategorisierung, numerische Werte, funktionale Zusammenhänge)
4. Lösungsverfahren zur Parametrisierung des Modells (z.B. Erfahrung des Auditors, analytisch, numerisch)
5. Referenzen

Simulation ist keine Methode im Sinne unserer Systematik. Insbesondere werden Monte-Carlo-Simulationen dazu verwendet, das Verhalten von physikalischen und mathematischen Systemen zu modellieren. Durch ihren stochastischen Charakter, basierend auf der Verwendung von Pseudozufallszahlen (indeterministisch), lassen sie sich von anderen (deterministischen) Simulationsmethoden unterscheiden.

Weiterhin ist festzuhalten, dass eine eindeutige Systematik, also die Kategorisierung einzelner Verfahren, auch im akademischen Bereich umstritten und ungelöst ist. Die überwiegende Zahl der hier betrachteten Verfahren lässt sich mehreren Kategorien zuordnen. So existieren Petri-Netze beispielsweise in vielfältigen Ausprägungen, von stark analytisch geprägten Ansätzen bis hin zu ausgeprägt quantitativen Ausprägungen (z.B. fluide oder farbige Petri-Netze). Ebenso lässt sich in ausgeprägt quantitativen Verfahren, wie beispielsweise kernelbasierten Regressionsverfahren, analytisch gewonnenes Wissen integrieren. Die Kategorisierung ist in diesem Text daher als subjektiv zu betrachten, basierend auf der Ausprägung des konkreten Verfahrens.

1.3 Analytische Methoden

1.3.1 Boolean logic Driven Markov Processes (BDMP)

1. *Beschreibung der Methode*

    Die generelle Idee bei BDMP ist die Verknüpfung jedes Blattes eines Fehlerbaumes mit einem Markov-Prozess. Die verwendeten Markov-Prozesse besitzen zwei Modi, sie sind in

Bezug auf die entsprechende Komponente oder das Teilsystem entweder aktiviert oder im Stand-by-Modus. Zu jeder Zeit hängt der Modus eines Markov-Prozesses von einer Booleschen Funktion anderer Prozesse ab, soweit er nicht als unabhängig definiert ist. Im Extremfall sind die Blätter eines Fehlerbaumes alle mit unabhängigen Markov-Prozessen verbunden.

2. *Eingabewerte der Methode*

Gemischt, funktionale Zusammenhänge (Fehlerbaum), Übergangswahrscheinlichkeiten (Markov-Modell).

3. *Ausgabe der Methode*

Numerische Werte (Ausfallwahrscheinlichkeiten); globales System, komponentenweise.

4. *Lösungsverfahren zur Parametrisierung des Modells*

Das Modell wird mittels des Werkzeugs FIGARO gelöst (Sequenzgenerierung und -quantifizierung).

5. *Referenzen*

Bouissou et al. (1991), Bouissou (2003).

1.3.2 Endliche Automaten (Finite State Machines)

1. *Beschreibung der Methode*

Endliche Automaten bestehen aus einer Menge an Zuständen, einem ausgezeichneten Startzustand, einem Eingabealphabet und einer Übergangsfunktion, die Eingabesymbole und den aktuellen Zustand auf den nächsten Zustand abbildet. Es gibt verschiedene Varianten von endlichen Automaten, unter anderem Mealy-Automaten (Ausgaben verbunden mit

Zustandsübergängen) und Moore-Automaten (Ausgaben beim Erreichen eines Zustands). Endliche Automaten können mittels Zustandsdiagrammen (State Diagrams) oder Zustandsübergangstabellen (State Transition Tables) dargestellt werden.

2. *Eingabewerte der Methode*

   Funktionale Zusammenhänge über das System, Prozessbeschreibung, Zustandsübergangsbedingungen.

3. *Ausgabe der Methode*

   Modell, spezifische Aussagen über das System in einem Zustand.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Auswertung der Zustandsübergangsfunktion (symbolisch).

5. *Referenzen*

   Gill (1962), Booth (1967), Hopkin and Moss (1976).

1.3.3 Ereignisablaufanalyse (Event Tree Analysis)

1. *Beschreibung der Methode*

   Ein Ereignisablauf ist eine graphische Repräsentation eines logischen Modells, welches mögliche Folgen eines initialen Ereignisses identifiziert und quantifiziert. Die Ereignisablaufanalyse gehört somit zu den induktiven Ansätzen der Zuverlässigkeitsbewertung. Das Ergebnis der Analyse wird graphisch in einem binären Baum dargestellt. Ein Zweig beschreibt das erfolgreiche Verhalten des Systems bei Eintritt des Ereignisses und der andere Zweig dessen Scheitern. Dadurch wird es möglich, verschiedene Pfade zu durchlaufen und eine Ereignissequenz zu identifizieren, die zu einem Systemausfall führt.

Die Ausfallwahrscheinlichkeit für einen Pfad lässt sich dabei aus den einzelnen Zweigwahrscheinlichkeiten des Pfades bestimmen. Die Gesamtwahrscheinlichkeit eines Systemausfalls wird aus den Wahrscheinlichkeiten derjenigen Pfade ermittelt, die zu einem Ausfall führen.

2. *Eingabewerte der Methode*

   Ausfallwahrscheinlichkeiten von Systemkomponenten oder -prozessen.

3. *Ausgabe der Methode*

   Ausfallwahrscheinlichkeit des Systems in Abhängigkeit von einem initialen Ereignis.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Lösung analytisch mittels Wahrscheinlichkeitstheoremen.

5. *Referenzen*

   Storey (1996).

1.3.4 Failure Occurrence Model (FORM) / Fault/Error Handle Model (FEHM)

1. *Beschreibung*

   Die Modelle werden innerhalb einer Methodik zur Beschreibung der Verlässlichkeit von komplexen, verteilten Systemen mittels stochastischer Petri-Netze verwendet. Verteilte Systeme besitzen mehrere, miteinander konkurrierende langsam und schnell ablaufende Aktivitäten. Zur Analyse werden die Systeme deshalb in Teilsysteme bezüglich einer Zeitskala dekomponiert. Das System wird im niedrig skalierten Zeitbereich mittels FORM und im hoch skalierten Zeitbereich durch FEHM beschrieben. Während FORM die Dynamik des Auftretens von Ausfällen untersucht, beschreibt FEHM die Details der Fehlererkennung

und -behandlung. FEHM ermöglicht eine Klassifizierung in transiente, intermittierende und permanente Fehler. Detaillierte Fehlerbehandlungsmethoden können mittels SPN (Stochastic Petri Nets) entwickelt und gelöst werden. Die Ergebnisse fließen zur Verlässlichkeitsbestimmung des Systems in das FORM ein.

2. *Eingabe*

   FORM: Funktionale Beschreibung (makroskopische Systemsicht)
   FEHM: Funktionale Beschreibung (mikroskopische Systemsicht).

3. *Ausgabe*

   Ausfallwahrscheinlichkeit des Gesamtsystems (FORM), zustandsbasierte Aussagen über das System (FEHM).

4. *Lösungsverfahren*

   Analytisch, Lösungsverfahren für Petri-Netze.

5. *Referenzen*

   Muppale and Lin (1996).

1.3.5 Fehlerbaumanalyse (Fault Tree Analysis)

1. *Beschreibung der Methode*

   Die Fehlerbaumanalyse ist eine Methode zur Berechnung der Wahrscheinlichkeit des Eintretens eines Ereignisses in Abhängigkeit von der Eintrittswahrscheinlichkeit oder Häufigkeit seiner verursachenden Ereignisse. Die Fehlerbaumanalyse gehört deshalb zu den deduktiven RCA-Methoden (*Root Cause Analysis*). An der Wurzel des Baumes befindet sich das unerwünschte Ereignis (TOP-Event). Jede Situation (Elementarereignis), welches

das TOP-Ereignis auslösen kann, wird mittels Boolescher Ausdrücke in den Baum eingefügt. Über die Auswertung der verknüpften Elementarereignisse wird die Zuverlässigkeit des Systems abgeleitet. Die Menge aller (mindestens erforderlichen) Elementarereignisse, die das unerwünschte Ereignis auslösen können, wird als (minimale) Schnittmenge (*Cut-Set*) bezeichnet. Der Vorteil der Fehlerbaumanalyse besteht in der leichten Beschreibbarkeit komplexer Systeme, jedoch wirkt sich die fehlende Modellierungsmöglichkeit der Fehler nachteilig aus, die von mehreren Modulen abhängig sind (gegenseitige Fehlerabhängigkeit).

2. *Eingabewerte der Methode*

   Textbeschreibung des unerwünschten Ereignisses, Eintrittswahrscheinlichkeit.

3. *Ausgabe der Methode*

   Quantitativ: Eintrittswahrscheinlichkeit eines Systemausfalls.
   Qualitativ: Bewertung der Fehler nach deren Verursacher (menschlich; aktive, passive Komponenten).

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Boolesche Algebra und Wahrscheinlichkeitstheoreme.

5. *Referenzen*

   Schneeweiss (1999), Johnson and Malek (1988).

1.3.6 Fehlerinjektion (Fault-injection)

1. *Beschreibung der Methode*

   Die Fehlerinjektion ist eine Methode, um die Zuverlässigkeit und Robustheit eines Systems

zu bewerten. Dabei werden folgende Kategorien unterschieden:

- Hardware-basierte Fehlerinjektion: Es wird die Zielplattform des Systems mit Hardwarefehlern versehen und anschließend das Verhalten des Systems auf der modifizierten Plattform untersucht.

- Software-basierte Fehlerinjektion: Es wird der Zustand des Systems mittels zusätzlicher Software abgeändert, um das Verhalten des Systems unter der Annahme des Eintritts eines Hardwaredefekts zu beobachten (Fehlerinjektion zur Laufzeit). Daneben kann auch eine Fehlerinjektion zur Kompilierungszeit durch Veränderung des Quellcodes einer Software erfolgen.

- Fehlerinjektion mittels Simulation: Es wird ein Modell des Systems simuliert.

- Hybride Fehlerinjektion: Kombination der drei anderen Techniken.

Gegenüber früheren Ansätzen einer Hardware-basierten Fehlerinjektion hat die Software-basierte Fehlerinjektion zunehmend an Bedeutung gewonnen, nicht zuletzt deshalb, weil sie eine Hardware-basierte Fehlerinjektion auf Gatterebene mittels eines FPGA (*Field-programmable Gate Array*) und Implementierungssprachen wie VHDL simulieren kann. Zudem bietet sie auch Vorteile wie geringe Entwicklungskosten, hohe Flexibilität und Portabilität. Die Methode der Fehlerinjektion wird auch im Robustheitstest bei der Systementwicklung angewendet, um das mögliche Verhalten des Systems unter fehlerhaften Bedingungen im realen Einsatz zu bewerten.

2. *Eingabewerte der Methode*

   Funktionale Zusammenhänge, Systemmodell, Black-Box und Grey-Box.

3. *Ausgabe der Methode*

   Numerisch: Abdeckungsmaße (*Coverage Measures*) zu möglichen Fehlerursachen, Systemleistung unter Fehlereinwirkung.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Simulation eines Systemmodells (Fehlerinjektion auf Modellebene), Ausführung des Testobjektes (Fehlerinjektion zur Laufzeit).

5. *Referenzen*

   Some et al. (2001), Clark and Pradhan (1995), Tang and Iyer (1996).

1.3.7 Petri-Netze

1. *Beschreibung der Methode*

   Petri-Netze sind als ein graphisches Modellierungswerkzeug weit verbreitet, um verteilte Prozesse und DES zu modellieren. Ein Petri-Netz ist ein bipartiter Graph, dessen Knoten aus Stellen oder Transitionen bestehen und dessen Kanten von Stellen zu Transitionen gehen oder umgekehrt. Zu Beginn werden die Stellen mit keinem oder beliebig vielen Marken (*Token*) belegt. Die Präsenz von Marken an einer Stelle indiziert, dass eine Bedingung oder ein Zustand im Prozess befriedigt oder eingetreten ist. Transitionen werden mit auftretenden Ereignissen verbunden. Eine Transition kann aktiviert werden, falls alle eingehenden Stellen mit mindestens einer Marke belegt sind. Nach der Aktivierung wird jeweils eine Marke von jeder eingehenden Stelle entfernt und zu jeder abgehenden Stelle hinzugefügt. Zu Petri-Netzen gibt es zahlreiche Erweiterungen, wie beispielsweise durch Hinzufügen von farbigen Marken (*Colored Petri Nets*) oder Nichtdeterminismus bei Transitionen (*Stochastic Petri Nets*). SAN (*Stochastic Activity Networks*) sind eine Verallgemeinerung stochastischer Petri-Netze, die zur Leistungs- und Zuverlässigkeitsanalyse verwendet werden.

2. *Eingabewerte der Methode*

   Funktionale Zusammenhänge, Wahrscheinlichkeiten (bei stochastischen Petri-Netzen).

3. *Ausgabe der Methode*

   Modell, Zustand des Systems zu einem beliebigen Zeitpunkt oder Ereignis.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Simulation mittels Schaltung, Transformation in eine Markov-Kette und deren analytischer Lösung.

5. *Referenzen*

   Petri (1962), Reisig (1992), Murata (1989), Movaghar (1985), Molloy (1982).

1.3.8 Prozessalgebra

1. *Beschreibung der Methode*

   Prozessalgebren werden dazu verwendet, das Verhalten von parallelen oder verteilten Systemen mit algebraischen Mitteln zu beschreiben. Dabei werden grundlegende (statische) Regeln, wie sequentielle, parallele und alternative Kompositionen, miteinander kombiniert, um einen Prozess zu beschreiben. Mittels dynamischer Regeln lässt sich die Ausführung einer Aktion in eine Prozessbeschreibung integrieren. Anhand von Verifikationsmethoden lässt sich zeigen, ob ein System eine Prozessbeschreibung erfüllt.

2. *Eingabewerte der Methode*

   Wissen über funktionale Zusammenhänge, Prozessbeschreibungen.

3. *Ausgabe der Methode*

   Analyse der Korrektheit, Fairness und teilweise auch Pünktlichkeit.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Verifikation.

5. *Referenzen*

   Hoare (1978), Milner (1989).

1.3.9 Reliability Centered Maintenance (RCM)

1. *Beschreibung*

   RCM ist ein industrieller Ansatz, bei dem die Identifizierung und Etablierung von Bedingungen zur Verbesserungen im betrieblichen Einsatz, der Wartbarkeit und der Finanzierung von Systemen im Vordergrund steht. RDM gleicht im Wesentlichen einer FMEA. Ziel dieser Methode ist es, dem Anwender die Möglichkeit zu geben, das Risiko der Ausfälle von Gerätschaften und Betriebsmitteln möglichst effektiv zu handhaben.

2. *Eingabe der Methode*

   Prozess- und Verhaltensbeschreibungen (System und menschlich).

3. *Ausgabe der Methode*

   Anweisungen zur Prozess- oder Verhaltensänderung.

4. *Lösungsverfahren*

   Qualitativ mittels Entscheidungstabelle.

5. *Referenzen*

Nowlan and Howard (1978).

1.3.10 Root-Cause-Analyse

1. *Beschreibung der Methode*

   Die Methode zur Fehlerursachenbehebung gehört zu einer Klasse von Problemlösungsmethoden, die darauf abzielen, die Wurzel oder Grundursache (*root cause*) eines Problems oder Ereignisses zu identifizieren. Die Anwendbarkeit der Methode beruht auf der Annahme, das Probleme am besten zu lösen seien, wenn man die Ursache korrigiert oder eliminiert, anstatt zu versuchen, die offensichtlichen Symptome des Problems zu beheben. Durch Ausführen korrigierender Maßnahmen an der Problemwurzel erhofft man sich eine Minimierung der Wahrscheinlichkeit des Wiederauftretens des Problems. Da das Wiederauftreten des Problems meist nicht durch eine einmalige Intervention unterbunden werden kann, wird die Methode auch als iterativer Prozess betrachtet, der zu einer kontinuierlichen Verbesserung führt. Zu den Root-Cause-Analysemethoden gehören unter anderem die Fehlerbaum- und Ereignisbaum-Analysen. Erstere werden auch als *deduktive*, letztere als *induktive* Analysemethoden bezeichnet. Weiterhin gibt es RCA-Methoden, beispielsweise die FMEA/FMECA, deren Schwerpunkt eher im qualitativen Bereich liegt (Beschreibung siehe qualitative Methoden). Es gibt verschiedene Werkzeuge, die unter dem Begriff der Fehlerursachenbehebung subsumiert werden, und deren Anwendungsmöglichkeiten in fünf verschiedene Bereiche eingeteilt werden: Betriebssicherheit, Herstellung, Prozess, Ausfall und System.

2. *Eingabewerte der Methode*

   System- oder Ereignisbeschreibung.

3. *Ausgabe der Methode*

   Quantitativ: Eintrittswahrscheinlichkeit.
   Qualitativ: Verbesserungsmaßnahmen.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Quantitativ: Boolesche Algebra und Wahrscheinlichkeitstheoreme.
   Qualitativ: Individuelle Maßnahmenbeschreibungen.

5. *Referenzen*

   Andersen and Fagerhaug (2006).

1.3.11 State-Charts

1. *Beschreibung der Methode*

   State-Charts sind eine Erweiterung von endlichen Automaten um Techniken, wie Verfeinerung in Subautomaten, Nebenläufigkeit und nichtdeterministischen Zustandsübergängen. Sie dienen der Spezifizierung und des Designs von reaktiven Systemen und werden hinsichtlich Hierarchie, Konkurrenz und Kommunikation organisiert.

2. *Eingabewerte der Methode*

   Aktivierende Bedingungen.

3. *Ausgabe der Methode*

   Modell, spezifische Aussagen über das System in einem Zustand.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Auswertung der Zustandsübergangsfunktion (symbolisch).

5. *Referenzen*

Harel and Pnueli (1985), Harel (1987).

1.3.12 Temporale Logik

1. *Beschreibung der Methode*

   Temporale Logik ist eine Form der Modallogik, bei der jede temporale Formel als wahr oder falsch gegenüber einem Modell der Welt interpretiert wird. Beispielsweise kann eine Formel gegenüber einem endlichen Automaten (*Finite State Machine*) validiert werden. Die Spezifizierung eines Systems in temporaler Logik wird gewöhnlich unterteilt in:

   - Betriebssicherheit (*Safety Conditions*): Bedingungen, die während des Systembetriebs nicht eintreten dürfen.
   - Lebendigkeit (*Liveness Conditions*): Bedingungen, die die Leistungen des Systems spezifizieren.
   - Fairness (*Fairness Conditions*): Bedingungen zur Lösung von nichtdeterministischen Spezifikationen.

   Die *Temporal Logic of Action* ist eine Variante der TL, bei der Systeme und deren Eigenschaften in derselben Logik repräsentiert werden. Durch die logische Implikation kann gezeigt werden, dass ein System seine Spezifikation erfüllt.

2. *Eingabewerte der Methode*

   Formeln über Systemeigenschaften in temporaler Logik, Spezifikation eines Weltmodells.

3. *Ausgabe der Methode*

   Wahrheitswerte (wahr, falsch).

4. *Lösungsverfahren zur Parametrisierung des Modells*

Model-Checking.

5. *Referenzen*

   Pnueli (1985), Lamport (1994).

1.3.13 Zuverlässigkeits-Blockdiagramme

1. *Beschreibung der Methode*

   Ein Zuverlässigkeits-Blockdiagramm ist ein funktionales Diagramm, um Ausfallabhängigkeiten innerhalb eines Systems zu modellieren. Dabei werden die Komponenten eines Systems in Blöcken dargestellt, die durch Linien miteinander verbunden sind. Ein Pfad durch ein RBD repräsentiert eine Menge an Komponenten, die im Betriebszustand voneinander abhängen. Jede Komponente kann deshalb aufgrund des Systemdesigns in mehreren Pfaden enthalten sein. Die grundlegenden Verbindungen zwischen Komponenten sind entweder seriell, parallel (redundant) oder eine Kombination davon. Fällt eine Komponente aus, wird die Verbindung darüber unterbrochen. Zuverlässigkeits-Blockdiagramme sind keine Abbildung der Systemstruktur, sie liefern hingegen quantifizierte Aussagen über die Zuverlässigkeit des Systems. In Zuverlässigkeits-Blockdiagrammen können keine Ausfallmodi unterschieden werden – eine Betrachtungseinheit ist entweder intakt oder defekt. Darüber hinaus wird vorausgesetzt, dass die Module statistisch unabhängig voneinander ausfallen.

2. *Eingabewerte der Methode*

   Beschreibung des Systems.

3. *Ausgabe der Methode*

   Numerisch: Zuverlässigkeit und Verfügbarkeit (MTTF, MTBF, etc.).

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Analytisch mittels Wahrscheinlichkeitstheoremen und Booleschen Operationen.

5. *Referenzen*

   Shooman (1987), Johnson and Malek (1988), Bream (1995).

1.4 Quantitative Methoden

1.4.1 Einführung

In der aktuellen Literatur werden überwiegend analytische Methoden zur Modellierung komplexer Computersysteme angewendet. Garg et al. (1998b) stellen ein analytisches Modell eines Softwaresystems zur Handhabung von Transaktionen vor. Auf Grund von Alterung nimmt die Bedienrate des analysierten Systems im Laufe der Zeit ab und es kommt zu Stillständen und Abstürzen der Software, welches zur Nichtverfügbarkeit führt. Weitere Ansätze beruhen auf Markov-Prozessen, um Softwaresysteme zu modellieren. Huang et al. (1995) beschreiben ein zeitkontinuierliches Markov-Ketten-Modell für ein hochverfügbares Telekommunikationssystem. Ausfallzeit und Kosten der Ausfallzeit werden von den Autoren als Funktion von Systemvariablen beschrieben. Die Annahme exponentiell verteilter, zeitunabhängiger Übergangsraten (Verweilzeit), die in Huang et al. (1995) aufgestellt wurden, werden in Dohi et al. (2000a) und Dohi et al. (2000b) abgeschwächt. Zudem wird dort ein Semi-Markov-Modell konstruiert. Auf diese Art finden die Autoren eine geschlossene Form für den optimalen Verjüngungszeitpunkt (*Rejuvenation Time)* des Systems. Garg et al. (1995) und Bobbio et al. (1999) entwickeln stochastische Petri-Netze, mit denen es möglich ist, den optimalen Zeitpunkt zum Neustart von Teilmodulen numerisch zu berechnen. Ein anderer häufig aufgeführter Ansatz ist die Konstruktion von Fehlerbäumen. Diese Methode beinhaltet die Durchführung einer vollständigen Aufzählung sämtlicher Systemstörungen Ulerich and Powers (1988). Falls eine Störung auftritt, kann diese mit der Grundursache (*root cause*) in Verbindung gebracht werden. Der Vorteil der Methode liegt darin, dass die Ursache der Störung identifiziert werden kann. Dieser Ansatz ist jedoch bei realen Systemen aufgrund untragbar hoher Komplexität nicht zu verwirklichen.

Eine besondere Stellung innerhalb der Gruppe der quantitativen Verfahren nehmen Black-Box-Verfahren ein, die fast ausschließlich auf empirisch erhobenen Daten arbeiten. Die Dynamik von Softwaresystemen kann in Form von Zeitreihen betrachtet werden, die die Evolution des Systems beschreiben. Als Beispiel einer solchen Zeitreihe kann die Ressourcenauslastung, zum Beispiel durch die Länge der Warteschlangen, Seitenzugriffsfehler oder Anzahl gelesener Seiten angeführt werden. Die Herausforderung dieses Ansatzes besteht in der automatisierten Erkennung der funktionalen Beziehung zwischen gemessenen Daten und Ausfällen, unabhängig von Datenrauschen, begrenzten Datenzugriffsmöglichkeiten und Datenverfügbarkeit sowie nichtlinearen Abhängigkeiten. Ein Ansatz zur Lösung dieser Problematik ist die nichtlineare, nichtparametrische Datenanalyse (Rumelhart et al. (1986), Hertz et al. (1991), Weigend et al. (1994), Bishop (1995)). Diese Modellierungstechnik adressiert einige Fragestellungen klassischer Analyse-techniken wie:

- nichtlineare Abhängigkeiten zwischen Systemvariablen zu modellieren
- der Umgang mit unvollständigen und verrauschten Daten
- der Umgang mit einer sich verändernden Systemdynamik
- der Umgang mit schwer interpretierbaren Daten (*Inconclusive Data*)
- Modellierung eines Systeme in der Produktivphase (*retrofitting*)

Insbesondere der Bedarf, ein System nach Fertigstellung zu modellieren (beispielsweise Daten während der Laufzeit über vorhandene Schnittstellen abzunehmen ohne das System durch zusätzlich zu erbringende Schnittstellen zu belasten), ist ein bedeutsamer Aspekt. Die meisten großen Softwaresysteme setzen Mechanismen zur Fehlererkennung während der Laufzeit ein. Diese ist typischerweise in Form von Grenzwertüberprüfungen einiger Parameter implementiert, die mit voreingestellten Werten verglichen werden. Sobald dieser vordefinierte gesunde Korridor verlassen wird, gibt das System einen Alarm aus. Die installierten Sensoren werden typischerweise während der Entwicklung des Quellcodes implementiert. In einigen Softwaresystemen stellen *Zähler* eine besondere Implementierung von Sensoren dar und werden deshalb in diesem Text synonym verwendet. Zusätzlich kann der Einbau von Sensoren in realen Systemen unerschwinglich teuer in Bezug auf die Systemlast werden, so dass ihre Implementierung weiter erschwert wird. Die Last, die durch Monitoring-, Speicherungs- und Wiederherstellungsoptionen erzeugt wird, kann leicht Ausmaße annehmen, die eine Abschaltung zur Folge haben.

1.4.2 Expertensysteme (wahrscheinlichkeitsbasiert)

1. *Beschreibung der Methode*

   Ein Modellierungsansatz, basierend auf Expertensystemen, wird von Weiss (Weiss (1999), Weiss (2001)) unternommen. Im *Timeweaver*-Projekt berechnet der Autor prädiktiv Ausfälle in Telekommunikationsausrüstungen. Der Autor identifiziert Logdateien als eine Quelle von Systemzustandsbeschreibungen. In seiner Fallstudie benutzt er genetische Algorithmen, um Regeln zu evolvieren, die auf zur Laufzeit gesammelten Logdateien eines Telekommunikationssystem basieren. Die gefundenen Regeln zeigten klare Vorhersagefähigkeiten. Ein beachtenswertes Ergebnis wird von Salfner et al. (2004) präsentiert. Die Autoren analysieren Logdateien als eine nützliche Quelle für Systemzustandsbeschreibungen und berechnen die in Logdateien enthaltene Entropie.

2. *Eingabewerte der Methode*

   Logdateien, Expertenwissen, statistische Auswertungen.

3. *Ausgabe der Methode*

   Regelbasiertes Modell.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Manuell oder mit Hilfe statischer Verfahren erstellte Regelsysteme.

5. *Referenzen*

   Weiss (1999), Weiss (2001), Salfner et al. (2004).

1.4.3 Failure Reporting, Analysis and Corrective Action System (FRACAS)

1. *Beschreibung*

   FRACAS ist ein *Framework* zur Unterstützung von Corrective-Action-Prozessen. Es wird im industriellen Bereich traditionell zur Analyse von technischen Systemen eingesetzt, lässt sich jedoch auch auf Software und Prozesse anwenden. Mittels des FRACAS-Prozesses wird die Verfügbarkeit eines Produktes oder Systems quantitativ und qualitativ beschrieben. Der Hauptzweck von FRACAS ist die Identifikation von Systemschwächen und die Einleitung geeigneter Gegenmaßnahmen. Zusätzlich können die Daten zur Berechnung numerischer Verfügbarkeitswerte genutzt werden.

2. *Eingabe der Methode*

   Daten aus Messungen während der Testphase, der Inbetriebnahme und des Betriebszustands.

3. *Ausgabe der Methode*

   Numerische Daten (Qualitäts- und Verfügbarkeitsanalyse), Informationen zur Fehlerbeseitigung und Reparaturanweisungen.

4. *Lösungsverfahren*

   Analytisch (z.B. Bestimmung von Ausfallzeiten), Maßnahmenkatalog.

5. *Referenzen*

   FRACAS (1999).

1.4.4 Fraktale Modellierung: Hölder-Exponent

1. *Beschreibung der Methode*

   Das chaotische Verhalten in Daten, die von Servern aus gewöhnlichen Büroumgebungen

stammen, wie beispielsweise Datei- und Druckservern, wird in Shereshevsky et al. (2001) untersucht. Die Autoren sammelten Daten zu zwei Variablen: Unbenutzte Speicherseiten (*Unused Memory Pages*) und freier Swap-Speicher. Anschließend berechneten sie aus den Daten den Hölder-Exponenten Parker and Chua (1989), der ein verbreitetes mathematisches Werkzeug zur Charakterisierung von chaotischem Verhalten in anderen Disziplinen, wie beispielsweise Medizinwissenschaften, Signalverarbeitung und Öko-nometrie darstellt. Die Autoren berichten über multifraktales Verhalten in beiden Zeitreihen und höhere Selbstähnlichkeit für höhere Arbeitslasten. Der fraktale Ansatz wurde weiterentwickelt von Crowell et al. (2002) und in ein Vorhersagesystem eingebaut Shereshevsky et al. (2003). Die Autoren berichten über einen Algorithmus, mit dem sie aufkommende Systemzusammenbrüche in 80% der Experimente richtig vorausberechnet haben. Jedoch geben sie keine Metriken an, wie beispielsweise residuale Fehler, *Area-under-Curve* (AUC), Präzision, Recall oder F-Measure, um ihren Ansatz vergleichbar zu machen.

2. *Eingabewerte der Methode*

   Hochfrequente Messdaten.

3. *Ausgabe der Methode*

   Numerische Werte.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Chaostheorie.

5. *Referenzen*

   Shereshevsky et al. (2001), Parker and Chua (1989), Crowell et al. (2002), Shereshevsky et al. (2003).

1.4.5 Lebensdatenanalyse (Life Data or Weibull Analysis)

1. Beschreibung

   Bei der Lebensdatenanalyse versucht der Anwender Vorhersagen über die Anpassung einer statistischen Verteilung an die Lebensdaten einer repräsentativen Stichprobe von Einheiten bezüglich der Lebensdauer eines Produktes zu treffen. Die parametrisierte Verteilung der Datenmenge kann dann dazu verwendet werden, um die Eigenschaften des Produktes zu schätzen, wie beispielsweise Zuverlässigkeit oder Ausfallwahrscheinlichkeit zu einer bestimmten Zeit sowie durchschnittliche Lebensdauer oder Ausfallraten.

2. *Eingaben*

   Messungen, repräsentative Stichprobe.

3. *Ausgaben*

   Numerische Werte (zu einem Zeitpunkt, stationär – MTTF und andere).

4. *Lösungsverfahren*

   Analytisch, Berechnung von Verteilungen.

5. *Referenzen*

   Nicht verfügbar.

1.4.6 Markov-Ketten

1. *Beschreibung der Methode*

   Markov-Ketten werden zur Modellierung von Systemen verwendet, deren Zustand eher

dynamisch als statisch ist (beispielsweise fehlertolerante Systeme mit Ersatzkonfigurationen). Jeder Zustand des Systems wird über seine Komponenten definiert, die entweder intakt oder ausgefallen sind. Der Zustandsraum des Systems definiert eine Markov-Kette mit endlichen Zuständen, falls folgende Bedingungen erfüllt sind:

- Es gibt nur eine endliche Anzahl von Zuständen.
- Die bedingte Wahrscheinlichkeit, das sich das System zukünftig in einem bestimmten Zustand befindet, hängt nur vom gegenwärtigen Zustand ab.
- Die Wahrscheinlichkeit des Übergangs von einem Zustand in einen anderen Zustand ist über die Zeit konstant.
- Für alle Zustände ist eine Menge an initialen Wahrscheinlichkeiten definiert.

Handelt es sich bei der Beobachtungszeit nicht um diskrete Zeitpunkte sondern um Zeitintervalle, so wird die Modellierung als Markov-Prozess bezeichnet. Die Übergänge von einem Zustand in einen anderen können in drei Kategorien klassifiziert werden:

1. Die Übergänge stellen Ausfallraten dar (sowohl für wiederherstellbare als auch nicht-wiederherstellbare Systeme).
2. Die Übergänge stellen Reparaturen dar, die das System von einem fehlerfreien in einen anderen fehlerfreien Zustand überführen.
3. Die Übergänge stellen Reparaturen dar, die das System von einem fehlerhaften, ausgefallenen Zustand in einen betriebsbereiten Zustand überführen.

Die Darstellung des Systems führt schnell zu einer großen Menge an möglichen Systemzuständen. Von den Werkzeugen werden Methoden zur Vereinfachung verwendet (Zusammenfassung und Abschneiden von Zuständen sowie funktionale Dekomposition).

2. *Eingabewerte der Methode*

Übergangswahrscheinlichkeiten.

3. *Ausgabe der Methode*

Modell, spezifische Einzelwerte zu einem beliebigen Zeitpunkt.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Analytisch (Lösung linearer Gleichungssysteme, die aus der Zustandsübergangsmatrix abgeleitet werden).

5. *Referenzen*

   Markov (1906), Howard (1971), Booth (1967), Johnson and Malek (1988).

1.4.7 Musterabgleich (Pattern-Matching)

1. *Beschreibung der Methode*

   Der Musterabgleich kann als ein rudimentärer Ansatz zur statistischen Textanalyse innerhalb des Data-Mining gesehen werden. Er wird im Kontext der Bewertung von Systemzuverlässigkeit als eine Methode zur Datenanalyse von aufgezeichneten Systemdaten verwendet (*Logfiles*). Dabei wird mittels eines regulären Ausdrucks ein Muster aus Zeichenketten (*Pattern*) beschrieben und anschließend systematisch nach der Anwesenheit dieses Musters im Datenbestand gesucht. Eine Verfeinerung dieser Methode mittels Hidden-Markov-Modellen wird zur Vorhersage von System- oder Komponentenausfällen in Salfner (2005) und Salfner et al. (2004) beschrieben.

2. *Eingabewerte*

   Numerisch: Systemdaten (Logdateien).
   Reguläre Ausdrücke.

3. *Ausgabewerte*

Boolesche Funktion: Wahr/Falsch im Sinne von Ausdruck gefunden oder nicht gefunden. Statistisches Modell.

4. *Lösungsverfahren zur Parametrisierung*

Technische Grammatiken / Parsing, Hidden-Markov-Modelle.

5. *Referenzen*

Peyton Jones (1987), Bensberg (2001), Salfner (2005), Salfner et al. (2004).

1.4.8 Nichtlineare Regressionsmodelle: Kernelbasierte Verfahren

Bekannte und vielfach eingesetzte kernelbasierte Verfahren sind radiale Basis-Funktionen (RBF) und Support-Vektor-Maschinen (SVM).

1. *Beschreibung der Methode*

Netzwerke aus radialen Basis-Funktionen (RBF) wurden erstmalig zur Lösung von Interpolationsproblemen verwendet. Hierbei handelt es sich um das exakte Legen einer Kurve durch eine Menge von Punkten (für eine Zusammenfassung siehe Powell (1987)). RBF wurden zur Durchführung von allgemeineren Approximationsaufgaben von Broomhead and Lowe (1988), Moody and Darken (1989) und Poggio and Girosi (1990) weiterentwickelt. Poggio and Girosi (1990) zeigen, dass RBF für ein Framework der Regularisierungstheorie gut geeignet sind.

Die Verallgemeinerungseigenschaft der RBF hängt hauptsächlich von Kernpunkten in Regularisierungstechniken, Lernalgorithmen und Initialisierungsheuristiken sowie der Architektur ab. Es gibt zu diesen Themenstellungen eine große Anzahl an Literaturquellen. Jedoch werden RBF typischerweise in Verbindung mit A-Priori-Fixed-Kernel-Funktionen verwendet, und den Auswirkungen von Mixture-Kernels auf die Modellqualität und -effizienz in der Parameteroptimierung wurde bisher wenig Aufmerksamkeit geschenkt.

Im Einzelnen konnte gezeigt werden, dass einige Datenverteilungen nicht sehr gut von den Gaußschen Funktionen approximiert werden können (Freitas (1999)) und dass abhängig vom vorliegenden spezifischen Problem die Anpassung der Transferfunktionen an die darunter liegenden Daten die Modellqualität signifikant verbessert werden kann (Hastie and Tibshirani (1996), Schoelkopf and Smola (2002)). Um die Einschränkungen, die mit den RBF verbunden sind, zu überwinden, wurden die universellen Basis-Funktionen (UBF) eingeführt, welche im folgenden Textabschnitt beschrieben sind, und innerhalb von Telekommunikationssystemen sowie auf Webserver-Farmen angewendet werden.

Die Wahl der Transferfunktion ist in allen Kernel-basierten Lernalgorithmen entscheidend, wie beispielsweise bei den RBF. Die Transferfunktion setzt vorrangig Wissen als *Beigabe* von empirischen Beobachtungen ein. Obwohl diese Frage viele Male gestellt worden ist, wurde den Auswirkungen domainspezifischer Transferfunktionen auf Generalisierung und Effizienz der Parameteroptimierung bisher nur wenig Aufmerksamkeit geschenkt. Nachforschungen über die Rolle der Transferfunktionen sind immer noch in einem frühen Stadium und es ist nur wenig bekannt über die Auswirkung von partikulären Transferfunktionen, die auf konkrete praktische Datenklassen angewendet wurden. Gaußsche Funktionen werden häufig benutzt, jedoch gibt es keinen Grund, sie a-priori und ausschließlich zu benutzen. Deshalb wurden die Eigenschaften von alternativen Aktivierungsfunktionen, wie beschränkte und unbeschränkte Abdeckung von Entscheidungsregionen, in Hoffmann et al. (2006) untersucht. Die Autoren schlagen ein allgemeines Framework zur Kombinierung von Transferfunktionen vor. Sie nennen dieses Framework *universelle Basis-Funktionen* (UBF). Dieses Framework ist allgemein genug, um zwei Mischungsarten (*Mixture Types*) zu beinhalten: a) Mischung verschiedener Transferfunktionen in einem Netzwerk (solche wie Gaußsche, sigmoidale, multiquadratische) und b) der glatte Transfer, der alle Übergangszustände von einer Transferfunktion zur anderen Transferfunktion (z.B. Gaußsche nach sigmoidal) beinhaltet. Letztere erlaubt die Selbstadaption der Kernelausprägung (*Kernels Shape*), abhängig von der vorhandenen Datencharakteristik. Die Autoren zeigen, dass UBF die multivariaten linearen Regressionstechniken und ebenso die RBF statistisch signifikant überbieten.

2. *Eingabewerte der Methode*

Zeitreihen.

3. *Ausgabe der Methode*

Nichtlineare Gleichungssysteme, funktionale Zusammenhänge zwischen Variablen.

4. *Lösungsverfahren zur Parametrisierung des Modells*

Stochastische, numerische Optimierungsverfahren teilweise in Verbindung mit analytischen Ansätzen, zum Beispiel Matrixinversionen.

5. *Referenzen*

Powell (1987), Broomhead and Lowe (1988), Moody and Darken (1989), Poggio and Girosi (1990), Poggio and Girosi (1990), Hastie and Tibshirani (1996), Schoelkopf and Smola (2002).

1.4.9 Regressionsmodelle Nichtlinear: Multi-Layer-Perceptrons (MLP)

1. *Beschreibung der Methode*

Eine weitere Klasse nichtlinearer Regressionsanalysen stellen Multi-Layer-Perceptrons dar. Ein Multi-Layer-Perceptron besteht aus einer Eingabeschicht, einer oder mehreren inneren Schichten und einer Ausgabeschicht, wobei jedes Neuron einer Schicht mit jedem Neuron der nächsten Schicht verbunden ist Rumelhart et al. (1986). Multi-Layer-Perceptrons sind vorwärtsbetriebene Netze, die nach einer festen Lernaufgabe trainiert werden. Sie stellen universelle Approximatoren dar, d.h. sie sind in der Lage, jede beliebige stetige Funktion zu approximieren Rumelhart et al. (1986). Beim Lernen oder Training erfolgt die Veränderung der Synapsengewichte meist nach dem sogenannten Backpropagation-Algorithmus.

2. *Eingabewerte der Methode*

Zeitreihen.

3. *Ausgabe der Methode*

   Nichtlineare Gleichungssysteme, funktionale Zusammenhänge zwischen Variablen.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Backpropagation, stochastische Optimierung.

5. *Referenzen*

   Rumelhart et al. (1986), Hertz et al. (1991), Weigend et al. (1994), Bishop (1995).

1.4.10 Regressionsmodelle: Linear Autoregressiv

1. *Beschreibung der Methode*

   In der Literatur zur Anwendung von linearen Modellierungstechniken auf Softwaresystemen dominieren Ansätze, die auf einer einzelnen oder einer begrenzten Anzahl von Variablen beruhen. Die meisten Modelle basieren auf Beobachtungen über die Arbeitslast, die Zeit, den Speicher oder die Dateitabellen. Garg et al. (1998a) schlagen ein zeitbasiertes Modell zur Erkennung von Softwarealterung vor. Zur Vorhersage der Ressourcenüberlastung in Betriebssystemen, wie zum Beispiel Unix, stellen sich Vaidyanathan and Trivedi (1999) ein arbeitslastbasiertes Modell vor. Li et al. (2002) sammeln Daten eines realen Webservers und konstruieren ARMA-Modelle (*Autoregressive Moving Average*), um die Systemalterung und den Ressourcenbedarf zu schätzen. Chen et al. (2002) schlagen ihr Pinpoint-System zur Überwachung des Netzwerkverkehrs vor. Dazu entwickeln sie eine Cluster-basierte Analysetechnik, um Ausfälle mit Komponenten zu korrelieren und um zu entscheiden, welche Komponente sich höchstwahrscheinlich in einem Fehlzustand befindet.

Die explizite Modellierung eines adäquaten Verjüngungszeitpunkts, der die Systemverfügbarkeit optimieren und gleichzeitig die Systemressourcen schonen könnte, würde eine Anpassung an sich ändernde Systemdynamiken erlauben Dohi et al. (2000a). Dieser Ansatz würde weiterhin eine ereignisgesteuerte Initialisierung vorbeugender Maßnahmen (*preventive maintenance*) ermöglichen. Um dieses Ziel zu erreichen, werden datenbasierte Methoden auf Modelle komplexer Computersysteme angewendet. Li et al. (2002) schlagen ein lineares ARMA-Modell vor, um die Ressourcenüberlastung eines Apache Webservers unter einer Linux-Umgebung vorher zu sagen, basierend auf Zeitreihenmessungen der Systemvariablen. Sie vergleichen ihren Ansatz mit einer Anzahl von linearen Regressionsmodellen, die in Castelli et al. (2001) beschrieben sind. Die Autoren deuten an, dass sie eine nichtlineare Beziehung zwischen den Daten gefunden haben und deshalb die Benutzung von nichtlinearen Modellierungstechniken vorschlagen. Jedoch geben sie keine weiteren Informationen zu den Fehlerraten, um ihren Ansatz vergleichbar zu machen. AR (*autoregressive*) und ARMA (*autoregressive moving average*) Modelle werden in jedem statistischen Lehrbuch beschrieben, z.B. Box und Jenkins (1970).

2. *Eingabewerte der Methode*

   Zeitreihen.

3. *Ausgabe der Methode*

   Lineare Gleichungssysteme.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Z.B. mittels Matrixinversion.

5. *Referenzen*

   Box und Jenkins (1970), Garg et al. (1998a), Vaidyanathan and Trivedi (1999), Dohi et al. (2000a).

1.5 Qualitative Methoden

Informations-Risikomanagement (*Information Risk Management*) ist ein Prozess zur Bewertung von Risiken, denen die Infrastruktur einer IT-Organisation ausgeliefert ist, und zur Entwicklung einer Strategie, um diese Risiken zu verwalten. Der Prozess des Risikomanagements wird typischerweise aufgeteilt in die Risikobewertung (*Risk Assessment* – RA) und Risikoverminderung (*Risk Mitigation* – RM). Bei ersterer werden potentielle schädliche Bedrohungen für das Informationssystem identifiziert, während bei letzterer eine Strategie zum Umgang mit den Bedrohungen entwickelt und implementiert wird. Neben den drei Haupteigenschaften der Informationssicherheit *Vertrauen*, *Integrität* und *Verfügbarkeit* ist die Bedeutung der Verfügbarkeit signifikant gestiegen.

Die qualitative Bewertung der Verfügbarkeit eines Systems geschieht typischerweise in Form einer Zusammenstellung von Fragebögen und basiert auf Kenntnissen zum System, Erfahrungen (frühere Leistung) und die Fähigkeit diese beiden Dinge funktionsmäßig zu integrieren. Das Ergebnis hängt sehr stark vom persönlichen Erfahrungsschatz des Auditors ab.

Es gibt eine Vielzahl von Methodiken zum Risikomanagement (z.B. COBIT – *Control Objective for Information and related Technology*, BS 25999, ISO 17799, ISO 13335, OCTAVE oder BSI IT-Grundschutz). Es wird häufig argumentiert, dass gegenwärtige Standards zur Verfügbarkeits-bewertung Grenzen aufzeigen, sobald die Verfügbarkeit bewertet werden soll. Dies ist der Tatsache geschuldet, dass die Methoden weder die komplexen Abhängigkeiten zwischen den Bestandteilen einer IT-Infrastruktur noch zwischen der IT-Infrastruktur und den oben aufgesetzt laufenden Diensten beachten und integrieren. Die Berücksichtigung dieser Abhängigkeiten wird meistens dem Urteilsvermögen des Assessors der RA-Phase überlassen.

1.5.1 BS/ISO 17799 (27002)

1. *Beschreibung der Methode*

   ISO 17799 ist eine Menge an Steuerelementen, um einen Best-Practice-Ansatz in der Informationssicherheit zu verfolgen. Es handelt sich im Wesentlichen um einen generischen Standard zur Informationssicherheit. Der Vorgänger BS 7799-1 existierte bereits viele Jahre

in verschiedenen Versionen, obwohl der Standard erst durch die Veröffentlichung der ISO (*International Standard Organisation*) im Dezember 2000 Anerkennung fand. Formale Zertifizierung und Akkreditierung wurden zur selben Zeit eingeführt. Der Standard wurde im Juni 2007 als ISO/IEC 27002:2005 von der ISO renummeriert.

2. *Eingabewerte der Methode*

   Nicht verfügbar.

3. *Ausgabe der Methode*

   Nicht verfügbar.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Nicht verfügbar.

5. *Referenzen*

   BS/ISO-17799 (2005).

1.5.2 BSI IT-Grundschutz

1. *Beschreibung der Methode*

   Bei einer Risikobewertung nach dem IT-Grundschutz (BSI (2006),Humpert (2005)) wird ein Soll-Ist-Vergleich zwischen empfohlenen und bereits realisierten Maßnahmen durchgeführt. Die dabei festgestellten fehlenden und noch nicht umgesetzten Maßnahmen zeigen die Sicherheitsdefizite auf, die es durch die empfohlenen Maßnahmen zu beheben gilt. Erst bei einem signifikant höheren Schutzbedarf muss zusätzlich eine ergänzende Sicherheitsanalyse durchgeführt werden. Hierbei reicht es aber in der Regel aus, die Maßnahmenempfehlungen der IT-Grundschutz-Kataloge durch entsprechende individuelle, qualitativ höherwertige

Maßnahmen zu ergänzen. Die IT-Grundschutz-Kataloge beinhalten die Baustein-, Maßnahmen- und Gefährdungskataloge.
Sie umfassen:

- Standard-Sicherheitsmaßnahmen für typische IT-Systeme mit "normalem" Schutzbedarf
- eine Darstellung der pauschal angenommenen Gefährdungslage
- ausführliche Maßnahmenbeschreibungen als Umsetzungshilfe
- eine Beschreibung des Prozesses zum Erreichen und Aufrechterhalten eines angemessenen IT-Sicherheitsniveaus
- eine einfache Verfahrensweise zur Ermittlung des erreichten IT-Sicherheitsniveaus in Form eines Soll-Ist-Vergleichs

2. *Eingabewerte der Methode*

   Beschreibung des Systems.

3. *Ausgabe der Methode*

   Maßnahmenempfehlungen nach dem IT-Grundschutzkatalog, bzw. weitere individuelle Maßnahmen.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Nicht verfügbar.

5. *Referenzen*

   BSI (2006),Humpert (2005).

1.5.3 Control Objectives for Information and Related Technology (COBIT)

1. *Beschreibung der Methode*

COBIT Sewera (2005) ist ein Modell von generell anwendbaren und international akzeptierten IT-prozessbezogenen Kontrollzielen (*Control Objectives*), die in einem Unternehmen beachtet und umgesetzt werden sollten, um eine verlässliche Anwendung der Informationstechnologie zu gewährleisten (ITGI (2007), Bitterli (2007)). COBIT definiert dazu sieben Kriterien, anhand denen IT-Ressourcen zur Erreichung von Geschäftszielen bewertet werden können, eingeteilt in die folgenden Gruppen:

- Qualität der IT: Effektivität und Effizienz.
- Sicherheit: Vertraulichkeit, Integrität und Verfügbarkeit.
- Ordnungsmäßigkeit: Zuverlässigkeit sowie Einhaltung rechtlicher Erfordernisse.

Die Verwaltung der IT-Ressourcen wird wiederum in vier IT-Domains unterteilt:

- Planung und Organisation: Strategische Ebene; bestimmt wie die IT zur Erreichung der Geschäftsziele beitragen kann.
- Beschaffung und Implementation: Realisierung der IT und Integration in den Geschäftsprozess.
- Betrieb und Unterstützung: Effektive Bereitstellung der gewünschten Dienstleistungen.
- Überwachung: Qualität der IT und Einhaltung von Kontrollzielen.

COBIT wurde vom internationalen Prüfungsverband ISACA (*Information Systems Audit and Control Association*) seit 1993 entwickelt und als erste Version Ende 1995 veröffentlicht. Im Mai 1998 erschien eine komplett überarbeitete und erweiterte Version mit 34 IT-Prozessen und 300 Kontrollzielen. Im Juli 2000 wurde COBIT in der dritten Version im Wesentlichen um Aspekte der IT-Governance im Rahmen von *Management Guidelines* erweitert. Darin sind auch die Kernziele und kritischen Erfolgsfaktoren sowie die messbaren Leistungsindikatoren aufgeführt, welche eine klare Überwachung der IT-Prozesse durch das Management ermöglichen. Das COBIT-Framework besteht aus verbreiteten, generell akzeptierten Praktiken (Best Practices), welche sicherstellen, dass die benutzte Informationstechnologie die Geschäftsziele abdeckt und dass die Ressourcen verantwortungsvoll eingesetzt und die Risiken entsprechend überwacht werden, was mit dem Begriff IT-Governance bezeichnet wird. Die Bewertung bestehender Prozesse basiert

dabei auf der subjektiven Betrachtung eines Auditors. COBIT ist die einzige umfassende Methode zur Unterstützung von IT-Governance auf allen Ebenen.

2. *Eingabewerte der Methode*

   Prozessbeschreibungen und -bewertungen. Subjektive Bewertung von Prozessreifegraden durch einen Auditor.

3. *Ausgabe der Methode*

   Best-Practice-Methoden, Kontrollziele.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Nicht verfügbar.

5. *Referenzen*

   Sewera (2005), ITGI (2007), Bitterli (2007).

1.5.4 Failure Mode Effect Analysis (FMEA)

Hier werden Failure Mode Effect Analysis (FMEA) sowie Failure Mode Effect Criticality Analysis (FMECA) beschrieben.

1. *Beschreibung der Methode*

   Die FMEA/FMECA dient zur Identifikation und Priorisierung von Risiken sowie zur Ermittlung von Abhilfemaßnahmen. Sie beinhaltet sowohl analytische als auch qualitative Maßnahmen. Mittels der FMEA werden Fehler nach der Häufigkeit ihres Auftretens, nach der Erkennbarkeit und der Schwere ihrer Konsequenzen bewertet. Der Begriff „Failure Mode" beschreibt die Art und Weise, wie ein Fehler auftreten oder etwas fehlschlagen kann.

In der „Effect Analysis“ werden die Konsequenzen dieser Fehler untersucht. Die FMEA wird während der Designphase eines Systems verwendet, um zukünftig auftretende Fehler zu vermeiden. Später wird die FMEA zur Prozesskontrolle vor oder während ablaufender Systemoperationen verwendet. Mittels der FMEA werden Tätigkeiten angestoßen, die die Anzahl der Fehler reduzieren oder Fehler eliminieren sollen, beginnend mit den Fehlern der höchsten Priorität. Zusätzlich beinhaltet die FMECA eine Kritikalitätsanalyse, die die Wahrscheinlichkeit des Eintretens von Fehlerarten in Abhängigkeit von ihrem Schweregrad tabellarisiert. Das Ergebnis sind hervorgehobene Fehlerarten mit hoher Auftrittswahrscheinlichkeit und schwerwiegenden Konsequenzen, deren Beseitigung den größtmöglichen Nutzen bringt. Die FMEA wiederum dient als Ausgangspunkt für die HF-PFMEA (*Human Factors Process Failure Mode and Effects Analysis*). Die HF-PFMEA wird dazu verwendet, Aufgaben oder Prozesse zu bewerten, die von menschlichem Handeln bestimmt werden. Ziel ist es, eine fundierte Methode zu etablieren, um die Häufigkeit und Schwere menschlichen Fehlverhaltens und der damit verbundenen Unfälle zu reduzieren.

2. *Eingabewerte der Methode*

FMEA/FMECA: Zusammenstellung in Textform von Fehlerarten, einschließlich deren Ursachen und Folgen.
FMECA: Bewertung mit Hilfe der Risiko-Prioritätszahl RPZ (Eintrittswahrscheinlichkeit * Gewicht der Folgen * Wahrscheinlichkeit des Nichtentdeckens).
HF-PFMEA: Prozessmodell.

3. *Ausgabe der Methode*

Tabellarische Bewertung des Eintrittsrisikos eines Ausfalls aufgrund des Fehlers.
Maßnahmenvorschläge und -beschlüsse zur Beseitigung der Fehlerursache.
FMECA: Restrisikoanalyse (erneute Berechnung der RPZ), Kosten-Nutzen-Analyse.

4. *Lösungsverfahren zur Parametrisierung des Modells*

Qualitativ (tabellarisch), Einschätzung des Risikos durch den Modellierer.

5. Referenzen

Stamatis (1995), McDermott et al. (1996).

1.5.5 IT Infrastructure Library (ITIL)

1. Beschreibung der Methode

ITIL (*IT Infrastructure Library*) Sewera (2005) ist ein herstellerunabhängiges Regelwerk der zentralen IT-Beratungsstelle der britischen Regierung (*Central Computer & Telecommunications Agency*) und wurde in den späten 80er Jahren im Auftrag der britischen Regierung als Antwort auf die wachsende informationstechnologische Abhängigkeit entwickelt. Seit dem Jahre 2000 erfolgt die Herausgabe der Buchreihe durch das OGC (*UK Office of Government Commerce*). Das ITIL-Framework ist in Form einer themenspezifisch unterteilten Bücherreihe erhältlich, in der die ineinander greifenden Prozesse für das IT-Service-Management (Service-Support und Service-Bereitstellung) definiert werden. In diesen Büchern beschreibt ITIL nicht nur die reine Lehre, sondern auch ein systematisches, professionelles Vorgehen für das Management der IT und ihrer Dienstleistungen. Die Bücher enthalten außerdem Richtlinien zu anderen Themenbereichen, wie Sicherheits- und Business Management von IT. Das Best-Practice-Framework ITIL besteht seit dem Jahre 2002 aus sieben Hauptbereichen (*Sets*) mit über 20 Einzelprozessen, die fast alle Aspekte eines heutigen IT-gestützten Geschäftsalltages umfassen und den Rahmen vorgeben, wie ein optimaler IT-Service innerhalb einer Firma aussehen und wie dieser implementiert werden könnte. ITIL baut eine Brücke zwischen den geschäftlichen Anforderungen einer Firma und der IT-Technologie. Im Einzelnen werden die folgenden Hauptbereiche definiert, denen auch je ein Band (Buch) aus der Dokumentationsreihe ([Offi00a]-[Offi03]) zugeordnet ist:

- Business Perspective (Geschäftliche Perspektive)
- Service Delivery (Planung und Lieferung von IT-Service)
- Service Support (Unterstützung und Betrieb des IT-Services)
- Security Management (Sicherheitsmanagement)
- ICT Infrastructure Management (Management der Infrastruktur)
- Application Management (Management der Anwendungen)

- Planning to Implement Service Management (Planung der Einführung des Service-Managements)

In den Büchern werden nicht nur die Teilaufgaben aus den einzelnen Bereichen detailliert behandelt, sondern auch die Abhängigkeiten zwischen diesen Bereichen.
Im September 2007 wurde die derzeit aktuelle Version 3 veröffentlicht, die einige strukturelle Änderungen gegenüber der Vorgängerversion enthält. Die Bibliothek besteht nun aus den folgenden Bänden:

- Service Strategy
- Service Design
- Service Transition
- Service Operation
- Continual Service Improvement (CSI)

Der Fokus von ITIL hat sich in der Version 3 von der bloßen Definition eines Ziels hin zur Beschreibung einer Vorgehensweise zur Erreichung dieses Ziels bewegt. Weiterhin wird ein Schwerpunkt auf Schlüsselrollen (*key roles*) sowie der klaren Abgrenzung von Verantwortlichkeiten gelegt und die Bedeutsamkeit der Kommunikation während des gesamten Lebenszyklus hervorgehoben. Die Verwendung von Prozess- und Prozessverbesserungs-Modellen (CSI) spielt dabei eine wesentliche Rolle ITIL (2007).

Das IT Service Management Forum (itSMF) International (Van Bon et al. (2006)) ist eine profitlose, gesetzliche Körperschaft, die eingerichtet wurde, um bei der Ausweitung, Unterstützung und Koordination der weltweiten itSMF-Community beizutragen. Sie dient der Erzeugung und Weiterverbreitung von Best-Practice-Ansätzen im IT-Service-management, unter anderem nach ITIL. Best-Practice wird hier als bewährte Aktivitäten oder Prozesse definiert, die in verschiedenen Organisationen verwendet werden. Es wurde dazu eine Werkzeugsammlung zur Selbstbewertung konstruiert, um die ITIL-Best-Practice-Anweisungen umzusetzen und zu erweitern.

2. *Eingabewerte der Methode*

System- und Prozessbeschreibungen.

3. *Ausgabe der Methode*

   Verbesserungsmaßnahmen (Empfehlungen, Best-Practice).

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Nicht verfügbar.

5. *Referenzen*

   Sewera (2005), Van Bon et al. (2006).

1.5.6 Lebenszykluskosten-Analyse (Life Cycle Cost Analysis)

1. *Beschreibung der Methode*

   Diese Methode berechnet die Kosten eines Systems über die gesamte Zeitspanne seiner Existenz. Der Lebenszyklus umfasst folgende Stadien: Planung, Forschung und Entwicklung, Produktion, Einsatz, Wartung, Ersetzungskosten, Veräußerung oder Entsorgung. Die Analyse der Lebenszyklus-Kosten eines Systems ist zusätzlich abhängig von anderen Zuverlässigkeitsanalysen, wie Ausfallraten, Ersatzhaltungskosten, Reparaturkosten und Herstellungskosten einzelner Komponenten.

2. *Eingabewerte der Methode*

   Nicht verfügbar.

3. *Ausgabe der Methode*

   Nicht verfügbar.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Nicht verfügbar.

5. *Referenzen*

   Riggs (1982).

1.5.7 Microsoft Operations Framework (MOF)

1. *Beschreibung der Methode*

   Microsoft stellt seinen Kunden für die Planung und Realisierung von IT-Projekten Anleitungen basierend auf dem Regelwerk ITIL zur Verfügung. Das MOF besteht aus den folgenden Komponenten:

   - Process Model: Prozessbeschreibungen zur Steuerung und Wartung von IT-Services
   - Team Model: Team-Rollen und effektive Organisation der Mitarbeiter
   - Risk Management Model: Risikobewertung und -steuerung.

   Die Aufteilung der IT-Prozesse und -infrastruktur eines Unternehmens geschieht dabei entlang der Quadranten *Optimizing, Changing, Operating und Supporting*. Jeder Quadrant enhält entsprechende *key sets of activity* bzw. Service-Management-Functions (SMF). Die Anleitungen dienen als Modell und Orientierungshilfe bei Design, Entwicklung, Deployment, Betrieb und Support Microsoft-basierter Lösungen. Ziel ist eine Bewertung und Verbesserung der eigenen IT-Services. Die Basis für diese Consulting-orientierten Produkte sind sowohl das Know-How von Microsoft selbst und deren Firmenkunden als auch die am Markt verfügbaren Best-Practices. Die Anleitungen sind in zwei komplementäre und eng miteinander verbundene Frameworks (Rahmenstrukturen) aufgeteilt: Das Microsoft Operations Framework (MOF; Sewera (2005)) und das Microsoft Solutions Framework (MSF). Während MSF bewährte Vorgehensweisen für das Planen, Entwerfen, Entwickeln und Bereitstellen erfolgreicher IT- Lösungen anbietet, adressiert MOF den kontinuierlichen Betrieb von Microsoft-basierten IT-Lösungen, d.h. IT-Service-

Management. Angewendet wurde MOF unter anderem im „Windows 2000 Datacenter Management Program“.

2. *Eingabewerte der Methode*

   Prozess- und Risikobeschreibungen.

3. *Ausgabe der Methode*

   Maßnahmeempfehlungen, Prozessbeschreibungen.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Nicht verfügbar.

5. *Referenzen*

   Sewera (2005).

1.5.8 Risikobewertung (Risk Analysis and Assessment)

1. *Beschreibung der Methode*

   Die Risikobewertung ist eine Phase innerhalb des Risikomanagements und setzt sich aus folgenden Abschnitten zusammen:

   - Identifizierung von Risiken: Es werden Bedrohungen und Beschädigungsmöglichkeiten identifiziert, die mit Risiken in Verbindung stehen. Entscheidend hierbei ist, die Liste der Risiken so umfassend wie möglich zu halten. Dabei kann ein Risiko hinsichtlich seines Ursprungs, seinen Auswirkungen und seines Auftretens (Zeit, Ort oder aus speziellen Gründen) sowie der Inanspruchnahme von Schutzmechanismen charakterisiert werden.

- Analyse relevanter Risiken: Die Risikoanalyse ist der Abschnitt einer Risikobewertung, in der das Risiko hinsichtlich seiner Natur und seinem Grad eingeschätzt wird. Die Risikoanalyse enthält eine Untersuchung der Risikoursachen, der positiven und negativen Konsequenzen, der Eintrittswahrscheinlichkeit und Faktoren der Beeinflussung sowie der Bewertung von existierenden Prozessen, die negative Konsequenzen minimieren und positive Konsequenzen verstärken. Der Grad eines Risikos wird mittels statistischer Methoden aus verschiedenen Quellen (Aufzeichnung von Systemdaten, technische Dokumentationen, ökonomische, technische und organisatorische Modelle) bestimmt. Daneben können auch Fragebögen und das Wissen von Experten für eine Risikoanalyse nützlich sein.

- Risikoevaluierung: In dieser Phase werden Entscheidungen getroffen, die einer angemessenen Behandlung des Risikos entsprechen. Die Entscheidungen hängen vom Risikograd sowie von den weiteren Nebenbedingungen ab (Eintrittswahrscheinlichkeit des schadhaften Ereignisses, dessen Konsequenzen und Auswirkungen).

Durch anschließende Risikobehandlung sowie Formulierung von Risikoakzeptanzkriterien wird der Prozess des Risikomanagements vervollständigt (Stonebumer et al. (2002)).

2. *Eingabewerte der Methode*

   Nicht verfügbar.

3. *Ausgabe der Methode*

   Nicht verfügbar.

4. *Lösungsverfahren zur Parametrisierung des Modells*

   Nicht verfügbar.

5. Referenzen

Stonebumer et al. (2002)).

1.6 Diskussion und Auswertung der Ergebnisse

Die untersuchten Methoden wurden nach den Kriterien

1. Eingaben,
2. Ausgaben,
3. Lösungsverfahren und
4. Anwendungsgebiete

bewertet.

Eingaben beinhalten

1. Messungen am System, hierzu zählen
   1.1. Zeitkontinuierliche Messungen (Zeitreihen),
   1.2. Kategorische Daten (z.B. Logdateien) oder
   1.3. Wahrscheinlichkeiten (z.B. Übergangswahrscheinlichkeiten in einen fehlerhaften Status).
2. Wissen über funktionale Zusammenhänge. Dazu gehören beispielsweise Kenntnisse über das Zusammenwirken einzelner Komponenten oder Details der Softwarespezifikation.

Ausgaben beinhalten:

1. Konzeptionelle Ausgaben wie analytische Modelle (z.B. Petri-Netze und Markovketten) und Regressionsgleichungen.
2. Numerische Werte und funktionale Zusammenhänge. Hierzu zählen konkrete Werte für Systemvariablen (z.B. Verfügbarkeit und Zuverlässigkeit) aber auch Kategorisierungen (z.B. fehlerhaft oder nicht fehlerhaft).

Lösungsverfahren beinhalten numerische und analytische Verfahren sowie methodisches Wissen, zum Beispiel eines Auditors.

Die betrachteten Methoden zur Verfügbarkeitsanalyse wurden in drei Kategorien unterteilt:

1. Analytische Methoden
2. Quantitative Methoden
3. Qualitative Methoden

Hierzu ist festzuhalten, dass die genaue Kategorisierung einzelner Verfahren, auch im akademischen Bereich, umstritten und ungelöst ist. Die überwiegende Zahl der betrachteten Verfahren lässt sich mehreren Kategorien zuordnen. So existieren Petri-Netze beispielsweise in vielfältigen Ausprägungen, von stark analytisch geprägten Ansätzen bis hin zu ausgeprägt

quantitativen (z.B. fluide oder farbige Petri-Netze). Ebenso lässt sich in ausgeprägt quantitativen Verfahren, wie z.B. Kernel-basierte Regressionsverfahren, analytisch gewonnenes Wissen integrieren. Die Kategorisierung in diesem Text ist daher als subjektiv zu betrachten, basierend auf der Ausprägung des konkreten Verfahrens.

In der zusammenfassenden Auswertung am Ende dieses Kapitels werden die Verfahrenskategorien hinsichtlich

- Steady-State- vs. Service-Verfügbarkeit,
- den Anwendungspotentialen,
- der Skalierbarkeit sowie
- dem Aufwand der Datenbeschaffung

diskutiert und beurteilt.

1.6.1 Analytische Methoden

Als notwendige Voraussetzung zum Einsatz analytischer Verfahren sind Kenntnisse bzw. Annahmen über den funktionalen Zusammenhang interner Abläufe und Prozesse des Systems notwendig. Teilweise können diese Annahmen durch Schätzungen oder gemessene Verteilungen von Eintritts- und Übergangswahrscheinlichkeiten bestimmter Ereignisse ergänzt oder korrigiert werden. Analytische Verfahren werden häufig auch als W*hite-Box*- oder G*rey-Box*-Verfahren bezeichnet, um zu verdeutlichen, dass beim Anwender ein ausgeprägtes Systemverständnis vorhanden sein muss, um eine sinnvolle Anwendung dieser Verfahren durchzuführen. Es erfolgt eine statische Betrachtung des Systems, konkret bedeutet dies, dass die Steady-State-Verfügbarkeit des Systems abgeschätzt werden kann, typischerweise jedoch nicht die Intervall-Call-Verfügbarkeit oder die Service-Verfügbarkeit. Eine fundamentale Schwäche analytischer Verfahren besteht darin, dass Systemänderungen nur mit größtem Aufwand verfolgt und im Modell abgebildet werden können. Zur Linderung dieser Problematik existieren Ansätze, die eine potentielle Dynamisierung der Verfahren durch Veränderung der zugrunde liegenden Wahrscheinlichkeitsverteilungen unterstützen sollen, zum Beispiel bei Petri-Netzen und Markov-Ketten. Andere Ergänzungen versuchen mittels Analyse der Differenzen aus Soll-Ist-Vergleichen fehlerhafte Komponenten zu identifizieren, siehe hierzu Frank (1990) und Patton et al.(1989). Das grundlegende Problem analytischer Verfahren besteht jedoch darin, dass die Komplexitätsgrade moderner Rechnersysteme, aufgrund kombinatorischer Vielfalt und Dynamik, nur unzureichend oder unter Einsatz erheblicher Ressourcen handhabbar sind. Dies ist insbesondere für die Ansätze temporaler Logik und Prozess-algebra festzustellen.

Vorteile analytischer Verfahren:

- Breites Anwendungsgebiet über vielfältige Industriezweige hinweg, zum Beispiel Verfügbarkeitsabschätzungen in der Luft- und Raumfahrt, dem Energiesektor sowie bei Rechner- und Softwaresystemen.

Nachteile analytischer Verfahren:

- Prohibitiver Aufwand bei komplexen Systemanalysen, schnell unbeherschbarer Aufwand bei realen Rechnersystemen,
- Sehr hohes Detailwissen über das System notwendig.

1.6.2 Quantitative Methoden

Die empirische wahrscheinlichkeitsbasierte Modellierung eines Systems eröffnet einen Ausweg aus dem Dilemma der kombinatorischen Explosion, welches analytische Verfahren einschränkt. Die grundlegende Idee besteht darin, ein System empirisch zu erfassen und funktionale Zusammenhänge anhand gewonnener Daten abzuleiten, siehe hierzu unter anderem Littlewood and Strigini (2000) und Hoffmann (2005). Die in dieser Studie untersuchten quantitativen Verfahren lassen sich wie folgt kategorisieren:

- Expertensysteme
- Fraktale Modellierung
- Lineare statistische Verfahren
- Nichtlineare statistische Verfahren

Expertensysteme würden typischerweise nicht als quantitatives Verfahren klassifiziert werden. Industrienahe, dokumentierte Einsatzbereiche zur Verfügbarkeitsbestimmung beruhen jedoch darauf, dass Expertensysteme basierend auf statistischen Auswertungen von Logdateien etabliert wurden. Die bekannten und dokumentierten Einsätze stammen aus den Jahren 1999-2001. Es ist zur Zeit unbekannt, in wie weit diese Verfahren breiten Einzug in industrienahe Anwendungen gefunden haben. Die Ergebnisse statistischer Auswertungen von Logdateien sind bislang volatil. Positive Fehlervorhersagequalitäten wurden von Weiss (1999) und Weiss (2001) dokumentiert. Sofern Logdateien nicht qualitativ hochwertigen Standards genügen, kann die Einbeziehung jedoch auch zu einer deutlich negativen Beeinträchtigung des Modellierungsergebnisses führen (Hoffmann (2005)).

Damit eine fraktale Datenanalyse erfolgreich eingesetzt werden kann, müssen umfangreiche Datenmengen mit geringem bis keinem Rauschen vorliegen (Fraser and Swinney (1986), Lapedes and Farber (1987)). Beide Voraussetzungen werden im realen Betrieb jedoch leicht verletzt, so dass der Einsatz dieser Verfahren bislang auf wenige, akademisch geprägte Einsatzgebiete beschränkt blieb, zum Beispiel Shereshevsky et al. (2001) und Crowell et al. (2002).

Empirische lineare und nichtlineare Modellierungstechniken werden im industriellen Umfeld vielfältig eingesetzt, beispielsweise zur Modellierung komplexer Produktionsanlagen (Neville (1998), Venkatasubramanian et al. (2003)), in der Raumfahrt (Smyth (1994a)) und zur Modellierung von Teilaspekten im Betrieb von Kernkraftwerken (Hines et al. (1999)). Die Anwendung auf Softwaresysteme ist eine junge Disziplin, die sich bislang darauf konzentriert, den linearen Verlauf bestimmter Systemparameter festzustellen, zum Beispiel der Speicherausnutzung (Vaidyanathan and Trivedi (1999)). Konkrete Aussagen zur Service-Verfügbarkeit werden in Hoffmann (2005) gemacht.

In quantitativen Verfahren gehen Messwerte über Sensoren ein, die im System angebracht sind. Hierzu zählen:

- Zeitreihen, zum Beispiel zur CPU Auslastung und Speichernutzung
- Kategorische, vorfallsausgelöste Daten (*event-triggered*), z.B. Logdateien
- Gegebenenfalls Kenntnisse über den funktionalen Zusammenhang interner Abläufe und Prozesse des Systems

Als Ausgabe liefern quantitative Verfahren:

- [Nichtlineare,] funktionale Zusammenhänge von Systemvariablen, insbesondere Service-Verfügbarkeit als Funktion bestimmter Systemparameter
- Zeitnahe, dynamische Betrachtung des Systems

Vorteile quantitativer Verfahren:

- Dynamische Systemmodellierung, damit auch die Möglichkeit zur Modellierung und Prognostizierung von Service-Verfügbarkeit
- Ausführung von Sensitivitätsanalysen, z.B. wie verhält sich die Service-Verfügbarkeit in Abhängigkeit von Systemlast, Speicherausnutzung und Zeit
- Punktuelle konditionale Wahrscheinlichkeiten (Berechnung von *conditional probabilities*)
- Bei entsprechender Vorverarbeitung kann die Analyse und Modellierung zeitkontinuierlicher

Daten (Zeitreihen) als auch kategorischer Daten integriert werden
- Methodenabhängig können Kenntnisse zum System in die Modellbildung integriert werden (Black-Box vs. Grey-Box vs. White-Box)

Nachteile quantitativer Verfahren:
- Bei stark belasteten Systemen wird die Datenerfassung zugunsten kritischerer Prozesse zurückgefahren.
- Während lineare Modelle analytisch gelöst werden können, stellt die Parametrisierung nichtlinearer Modelle eine Hürde dar. Die Parameterschätzung erfolgt überwiegend mit rechenintensiven numerischen Verfahren, beispielsweise mittels stochastischen Optimierungsverfahren.
- Validierung und Anpassung an sich ändernde Systemdynamiken; die sichere Erkennung eines neuen Systemstatus und damit die Anpassung des eingesetzten Modells ist Gegenstand aktueller Forschung.
- Die Auswahl geeigneter Variablen aus der Vielzahl verfügbarer Variablen ist nicht trivial und Gegenstand aktueller Forschung.

1.6.3 Qualitative Methoden

Die qualitative Systemanalyse erfolgt meist in Form einer Befragung oder Inspektion, bzw. prüft die Einhaltung allgemein akzeptierter Regelwerke (*Best-Practices*) zur Handhabung von Rechnersystemen oder Prozessen. Neben technischen Aspekten können je nach Methode, auch Prozessabläufe aufgenommen und ausgewertet werden (z.B. bei COBIT). In die Bewertung der aufgenommenen Daten fließen überwiegend Kenntnisse über den funktionalen Zusammenhang interner Abläufe und Prozesse des zu untersuchenden Systems ein, sowie die subjektive Bewertung und Erfahrung des ausführenden Auditors. Erklärtes Ziel ist zwar, ein verifizierbares und quantifizierbares Ergebnis zu erzeugen, letztlich hängen die Ergebnisse jedoch in starkem Maße vom ausführenden Auditor oder Planer ab. Die Ergebnisse einer qualitativen Systemanalyse sind üblicherweise:
- Zuordnung zu einer bestimmten Risiko- oder Verfügbarkeitsklasse
- Numerische Referenzwerte, ggf. Benchmarking mit vergleichbaren Industrien
- Ggf. Ableitung konkreter Handlungsanweisungen zur Erreichung bestimmter Ziele

Vorteile qualitativer Methoden:

- Technische als auch nicht-technische Aspekte (z.B. physische Zugangskontrollen, Vorhandensein von Handlungs- und Notfallplänen) können integriert werden
- Benchmarking innerhalb vergleichbarer Industrien möglich

Nachteile qualitativer Methoden:

- Aufwändige, zeit- und kostenintensive Datenbeschaffung und Durchführung: Die Zertifizierung eines Unternehmens oder Systems nach einer bestimmten Methode nimmt leicht mehrere Personenmonate in Anspruch und ist mit entsprechenden Kosten verbunden.
- Betrachtung einer Momentaufnahme des Systems (*Snapshot*) erlaubt Adaptierung, Kontrolle und Einleitung von Steuerungsmaßnahmen nur mit Zeitverzögerung
- Statische Betrachtung des Gesamtsystems zum Zeitpunkt der Analyse

Die Ableitung einer Steady-State-Verfügbarkeit, einer Intervall-Call-Verfügbarkeit oder einer Service-Verfügbarkeit (siehe hierzu Definitionen) ist m.E. mit den untersuchten Verfahren nicht zu erreichen. Es bleibt zu prüfen, ob Einzelerhebungen bestimmter Control-Objectives von COBIT innerhalb quantitativer Verfahren zu einer Qualitätsverbesserung dieser Modelle führen können. Diese Problematik fällt in den Bereich Variablenselektion, der Gegenstand aktueller Forschung ist Hoffmann (2005).

1.6.4 Zusammenfassung

Steady-State-Verfügbarkeit bezeichnet die statische Verfügbarkeit des betrachteten Systems während seiner Gesamtlebenszeit. Durch Messungen über die gesamte Lebenszeit des Systems könnte die Steady-State-Verfügbarkeit empirisch ermittelt werden, jedoch ist dies nicht praxistauglich. Zur Abschätzung der Steady-State-Verfügbarkeit werden daher überwiegend analytische Verfahren (z.B. Petri-Netze) und teilweise unterstützend qualitative Verfahren (z.B. COBIT) eingesetzt.

Service-Verfügbarkeit oder Intervall-Call-Verfügbarkeit bezeichnet die Systemverfügbarkeit in einem beliebigen Zeitintervall, welches signifikant kleiner sein kann als die Gesamtlebenszeit des Systems. Die Service-Verfügbarkeit kann beispielsweise in Abhängigkeit weiterer Systemvariablen dargestellt werden. Hierzu werden überwiegend quantitative, empirische Verfahren eingesetzt, wie beispielsweise lineare, multivariate Regressionsanalysen oder Kernel-basierte, nichtlineare Regressionsverfahren. Das Einsatzgebiet dieser Verfahren ist gebietsübergreifend und in großen

Bereichen der industriellen Modellierung bereits weit fortgeschritten (z.B. Modellierung von Industrieanlagen, Genomanalysen, Finanzanalysen). Der Einsatz im Bereich Modellierung von Service-Verfügbarkeit ist noch jung und überwiegend ein Forschungsthema. Im Gegensatz zu analytischen und qualitativen Verfahren liegen erst wenige dokumentierte Ergebnisse vor. Basierend auf dem erfolgreichen Einsatz quantitativer Verfahren in einer Vielzahl von Industrien und Anwendungsgebieten und basierend auf eigenen Forschungen, gehen wir jedoch von einer stark steigenden Bedeutung quantitativer Verfahren im Bereich Verfügbarkeitsanalysen aus.

1. Anwendungspotentiale: IT-Ebene vs. Service-Ebene

Diese Fragestellung ist eng verknüpft mit der Frage nach der abzubildenden Verfügbarkeit: Steady-State- vs. Service-Verfügbarkeit. Die Mehrzahl der untersuchten analytischen und quantitativen Methoden wird gegenwärtig auf der IT-Ebene zur Abschätzung von Steady-State-Systemverfügbarkeit und Zuverlässigkeit eingesetzt. Der Einsatz qualitativer Verfahren kann zwar Ansatzweise die Betrachtung von IT-Ebene und Service-Ebene integrieren (z.B. COBIT), vermag jedoch keine aussagekräftigen Parameter liefern, die den Vergleich von Soll- und Ist-Verfügbarkeit ermöglichen würde. Als Verfahren zur Modellierung von Service-Verfügbarkeit kommen nach unserem Wissen aktuell nur quantitative Verfahren in Frage.

2. Skalierbarkeit

Analytische Verfahren stoßen bei steigender Komplexität des zu analysierenden Systems schnell an ihre Kapazitätsgrenzen. So sind Prozessalgebra und temporale Logik gut geeignet, um zu verifizieren, ob ein System seiner Prozessbeschreibung oder Spezifikation genügt, jedoch sind die entstehenden Modelle nur für Systeme mit wenigen Systemzuständen handhabbar. Industriell eingesetzte Systeme können mit diesen Verfahren nur unter Einsatz erheblicher Ressourcen verifiziert werden. Der Einsatzbereich dieser Verfahren ist daher auf Nischen beschränkt, die entweder die notwendigen Ressourcen aufbringen (z.B. nukleare Kraftwerkstechnologie, Flugkontrollsysteme) oder wenig komplexe Systeme analysieren. Quantitative Verfahren sind aufgrund ihres empirischen Charakters für die Modellierung komplexer Systeme prädestiniert. Problematisches Skalierungsverhalten wird hier weniger vom Verfahren vorgegeben als durch die eingesetzten Lösungsverfahren zur Modellparametrisierung und durch die Datenbeschaffung und Vorverarbeitung. Siehe dazu auch den folgenden Absatz „Aufwand der Datenbeschaffung“. Bei linearen, multivariaten Verfahren kommen typischerweise analytische Lösungsverfahren (z.B. Matrixinversion) mit einer Komplexität von $O(n^3)$ zum Einsatz. Bei nichtlinearen Verfahren (z.B. Kernel-basierte Verfahren wie Radiale-Basis-Funktionen oder Support-Vektor-Maschinen) kommen

überwiegend stochastische Optimierungsverfahren zum Einsatz, die einen erheblichen Rechenbedarf erzeugen.

3. Aufwand der Datenbeschaffung

Bei analytischen Verfahren beschränkt sich die Datenbeschaffung überwiegend auf die Abschätzung von Übergangswahrscheinlichkeiten. In vielen Fällen wird vereinfachend von Gaußschen Normalverteilungen ausgegangen. Die Datenerhebung bei qualitativen Verfahren ist aufwändig und kann je nach System mehrere Personenmonate in Anspruch nehmen. Zu beachten ist hierbei, dass lediglich ein Schnappschuss des Systems aufgezeichnet wird. Die Datenbeschaffung bei quantitativen Verfahren kann ein limitierender Faktor sein. Hierbei ist zu berücksichtigen, dass bei industriell eingesetzten Systemen typischerweise nur zwei Datenschnittstellen zur Verfügung stehen, das Betriebssystem und ggf. die Applikation selbst. In Situationen hoher Systemlast werden häufig Protokollmechanismen außer Kraft gesetzt, um Ressourcen (in diesem Fall CPU-Zeit, Plattenzugriffe, Speicher, Netzwerkauslastung und andere) einzusparen. Dies verhindert dann eine lückenlose Datenaufzeichnung. Während die quantitativen Verfahren selbst größtenteils technologisch ausgereift sind und in vielfältigen Szenarien bereits industriell zum Einsatz kommen, besteht bei der Datenaufbereitung und der Selektion wichtiger Eingangsvariablen noch Forschungsbedarf.

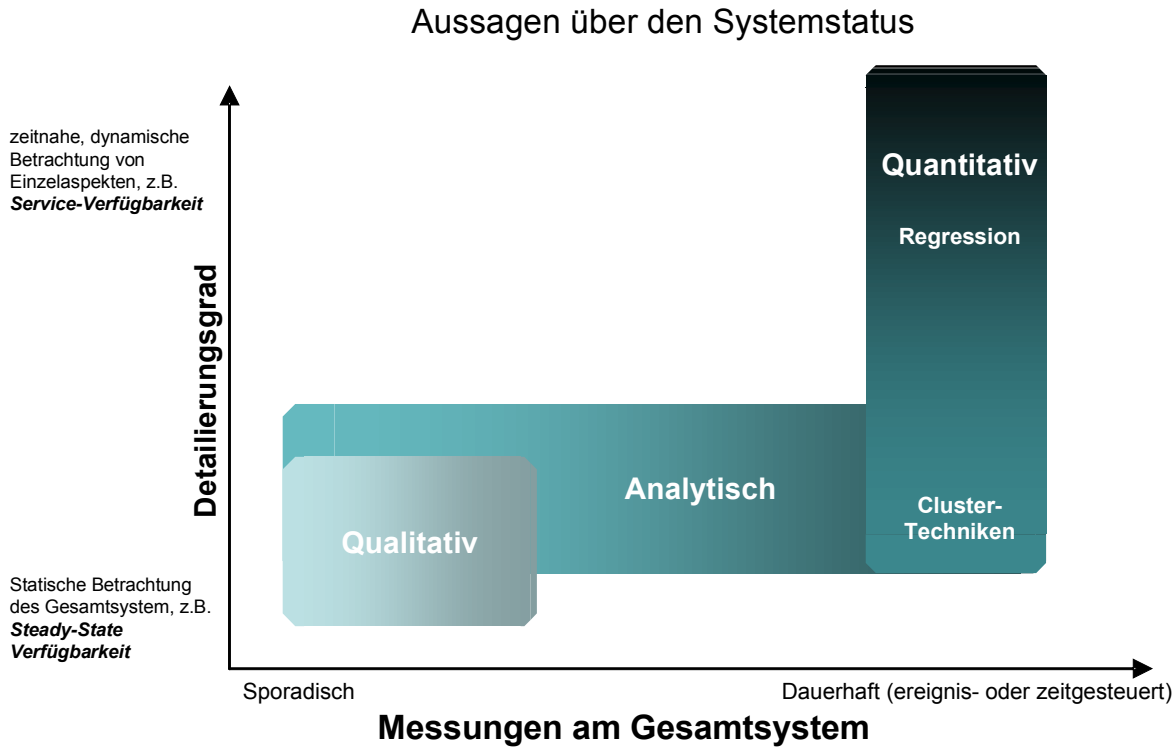


Abbildung 2: Je nach gewünschtem Detaillierungsgrad der Aussagen über den Systemstatus ist die Anwendung unterschiedlicher methodische Ansätze angezeigt. Siehe hierzu auch den vorangegangen Abschnitt.

2 Werkzeuge zur Verfügbarkeitsermittlung

2.1 Auswahl und Bewertungskriterien

Die untersuchten Werkzeuge sind in zwei Hauptkategorien unterteilt, basierend auf: quantitativen (einschließlich analytischer Methoden, Simulations- und Benchmarking- bzw. Prüfungs-Methoden) und qualitativen Methoden. Die Beschreibung der Werkzeuge erfolgt in einer Übersicht, die die folgenden Punkte beinhaltet:

- Quellen / Referenzen: Öffentlich verfügbares Material zu den Werkzeugen, darunter wissenschaftliche Abhandlungen, Gesetzesvorlagen, Fachbeiträge, Dokumentationen, Benutzerhandbücher und gegebenenfalls Homepages zum Projekt oder zum Werkzeug.
- Projektstatus: Status der Entwicklung des Werkzeugs; Prüfung, ob das Tool noch stetig weiterentwickelt wird.
- Lizenztyp: Auflistung aller bekannten Lizenztypen, dazu gehören die kommerzielle (möglicherweise mit Preisangaben) und die akademische sowie eine Evaluierungs-Lizenz.
- Allgemeiner Zweck: Kurze Übersicht der Hauptfunktionen, der verwendeten Modelle und möglicher Lösungen.
- Plattformen: Auflistung bekannter, verfügbarer Plattformen, auf denen das Werkzeug genutzt werden kann; zusätzliche Informationen über die Infrastruktur, wenn beispielsweise ein Werkzeug auf einer Datenbank arbeitet, werden die unterstützten Datenbanken angegeben.
- Modell:
  - Modellklasse: Einordnung der Werkzeuge in eine oder mehrere der folgenden Kategorien: quantitativ (analytisch, Simulation oder Benchmarking) und qualitativ; Angabe, auf welcher Ebene das Werkzeug anwendbar ist, z.B. IT- oder Geschäftsprozess- bzw. Service-Ebene.
  - Modelltypen: Auflistung der primär verwendeten Modelltypen des Werkzeugs, wie beispielsweise Markov-Ketten, Petri-Netze, Fragebögen, Workflows und Component Failure Impact Analysis.
  - Modellbeschreibung: Kurzer Überblick über das betrachtete Modell, einschließlich einer knappen Beschreibung und der Haupteigenschaften.
  - Modelleingabe: Auflistung aller notwendigen Eingaben, z.B. Knoten, Kanten und

Ausfallarten.

- Modellausgabe: Angabe der verfügbaren Ergebnisse der Modellanalyse, z.B. Verfügbarkeitsmetriken, MTTF und MTTR.
- Schnittstellen: Beschreibung der technischen Schnittstellen des Werkzeugs.
  - Eingabeschnittstellen: Auflistung aller Möglichkeiten der Eingabe des Werkzeugs.
  - Ausgabeschnittstellen: Auflistung aller Möglichkeiten zur Generierung der Analyseergebnisse und -berichte.
- Anwendungsfälle: Auflistung der Anwendungsfälle, in denen eine Verfügbarkeitsanalyse unter Verwendung des betrachteten Werkzeugs durchgeführt wurde.
- Annahmen und Einschränkungen: Mögliche Beschränkungen, die bei der Nutzung des Werkzeugs auftreten können, wie Skalierbarkeit oder Sicherheit.

Am Ende der folgenden Beschreibungen aller Werkzeuge werden diese, bezüglich der oben genannten Kriterien und Kategorien, verglichen und eingestuft.

2.2 Quantitative Werkzeuge

ACARA

1. *Quellen / Referenzen*

   ACARA-2 (2007): http://www.openchannelfoundation.org/projects/ACARA_II
   Stalnaker (2007): http://naca.larc.nasa.gov/search.jsp

2. *Projektstatus*

   ACARA wurde von einer Gruppe von Ingenieuren am Glenn-Research-Center der NASA in Cleveland (Ohio) entwickelt. Eine aktuelle und verbesserte Version ist ACARA-2.

3. *Lizenztyp*

   ACARA-2: Lizenz der Open-Channel-Foundation, California Institute of Technology.

4. *Allgemeiner Zweck*

   ACARA (Availability Cost and Resource Allocation) ist ein Programm zur Analyse von Verfügbarkeit, Lebenszyklus-Kosten und Ressourcenplanung für periodisch reparierte Systeme. ACARA-2 verwendet eine Kombination der Exponential- und Weibull-Verteilung, um die Nutzungsdauer jeder Systemkomponente zu simulieren. Die Ersetzung jeder fehlerhaften Komponente wird simuliert, um die Systemleistungsfähigkeit zu optimieren und dennoch die Einschränkungen zu beachten, die mit der Herstellung der Komponenten und verfügbaren Ressourcen (Kapazität der Versorgungsfahrzeuge, Vorort-Ersatz, Arbeitskräfte) verbunden sind. Diese Einschränkungen werden vom Benutzer festgelegt. ACARA evaluiert die Verfügbarkeit des Systems mittels einer Blockdiagramm-Darstellung des Systems auf jedem Leistungsniveau.

5. *Plattformen*

Windows 98 oder neuer.

6. Modell

6.1. *Modellklasse*

Quantitativ – analytisch, IT-Ebene.

6.2. *Modelltyp*

Statistische Monte-Carlo-Methoden, Weibull- und Exponentialverteilung, Blockdiagramme, Modelle zum frühzeitigen und zufälligem Ausfall sowie durch Verschleiß.

6.3. *Modellbeschreibung*

ACARA modelliert folgende Ereignisse:

- Zeit bis zum Ausfall (*Time to Failure*): Für jeden Block wird mittels Weibull-und Exponentialverteilung und den oben beschriebenen Ausfallmodellen die Zeit bis zum Ausfall berechnet. Die Modelle werden mittels benutzerdefinierten Parametern angepasst, um die Ausfallcharakteristik jedes Blocks zu approximieren.

- Ausfallzeit (*Down Time*): ACARA schätzt die Zeit vom Ausfall eines Blockes bis zur Beschaffung eines Ersatzblock. Falls ein Ersatzblock lokal vorhanden ist, wird dieser Block sofort eingesetzt und die Ausfallzeit hängt nur von der MTTR ab. Falls der Ersatzblock nicht lokal vorhanden ist, plant ACARA die Ersetzung in Abhängigkeit von den Produktions- und Lieferkapazitäten des Blocks. ACARA überprüft auch die Bedingungen, wann ein ausgefallener Block ersetzt werden kann.

- Reparaturzeit (*Time to Repair*): Nachdem alle Bedingungen erfüllt sind, um den

ausgefallenen Block auszutauschen, schätzt ACARA die benötigte Zeit, um diesen Block zu ersetzen. Die Reparaturzeit hängt von der MTTR des jeweiligen Blockes ab. Zusätzlich werden drei Aspekte für den Austausch berücksichtigt: Wartungsteam, Ausrüstung und Robotertechnik.

6.4. *Modelliertes System*

Wiederherstellbare Systeme.

6.5. *Modelleingabe*

Die MTTR für jeden Block werden mittels einer Tabelle eingegeben (*Repair Time and Personnel Quantities Input Table*). Weiterhin können vom Benutzer Simulationsparameter (Simulationsdauer, Periodenlänge und Dauer der Ersatzbeschaffung) und Kostenparameter (Lohn, Transport, Ausrüstung und Robotertechnik) in die Software eingegeben werden. Bei den Ausfallmodellen gilt das für die Eingabe einzelner Parameter.

6.6. *Modellausgabe*

Die Modellausgaben gliedern sich in vier Teilbereiche:

- Leistungsfähigkeit:
  Kapazität-Zeit-Diagramm, Zustandsverfügbarkeit, Gesamtverfügbarkeit, äquivalente Verfügbarkeit, kumulative Verfügbarkeit, Zuverlässigkeit, kontinuierliches Zustandsverhalten.

- Ausfall und Reparatur:
  Ausfall-WDF (Wahrscheinlichkeitsdichtefunktion), kumulative WDF; Ausfallhäufigkeit in einer Periode und über alle Perioden, frühzeitige, zufällige und verschleißabhängige Ausfälle in einer Periode; Reparatur-WDF, Reparaturhäufigkeit in einer Periode und über alle Perioden, Kritikalität.

- Kosten für den Lebenszyklus:
  Kosten der Hardware, des Transports, der Mannschaft, der Ausrüstung und der Robotertechnik sowie die Gesamtkosten.

- Verteilung der Betriebsmittel (*Resource Allocation*):
  Depot-Bestand, ausgelieferte Hardware, Benutzung der Ressourcen.

6.7. Schnittstellen

6.7.1. Eingabe

Die Eingabe in ACARA erfolgt manuell über das Terminal.

6.7.2. Ausgabe

Die Ausgabe erfolgt graphisch oder in Textdateien.

7. Anwendungsfälle

Das Werkzeug wurde innerhalb der NASA entwickelt und vermutlich auch eingesetzt. Weitere Anwendungsfälle sind nicht bekannt.

8. Annahmen und Einschränkungen

ACARA nimmt an, dass die Wartungsarbeiten gleichzeitig geschehen.

ARIES

1 Quellen / Referenzen

Balakfushnan, Raghavendra (1990)
Makam, Avizienis (1982)

Makam et al.(1982)
Ng, Avizienis (1977)

2 Projektstatus

Zuerst wurde ARIES von Ng, Avizienis (1977) entwickelt, anschließend als ARIES 81 von Makam, Avizienis (1982) sowie ARIES 82 von Makam et al.(1982) modifiziert. Erweiterungen wurden in Balakfushnan, Raghavendra (1990) vorgeschlagen, jedoch ist nicht bekannt, ob sie implementiert wurden.

3 Lizenztyp

Die letzte bekannte Implementierung stammt aus dem Jahre 1982. Der Lizenztyp ist unbekannt.

4 Allgemeiner Zweck

Zuverlässigkeits- und Lebenszyklus-Analyse für fehlertolerante Systeme.

5 Plattformen

Es handelt sich dabei um APL- und C-Implementierungen.

6 Modell

6.1. Modellklasse

Analytisch, IT-Ebene.

6.2. Modelltyp

Homogene Markov-Modelle, gelöst mittels Lagrange-Sylvester-Interpolation.

6.3. *Beschreibung*

Das System kann über eine Zustandsübergangsmatrix oder als Serie von unabhängigen Teilsystemen beschrieben werden, welche jedes für sich identische Module enthält, die entweder aktiviert sind oder als Ersatzmodul dienen. Das Programm benutzt zur Lösung eine Technik der Matrixtransformation, die von unterschiedlichen Eigenwerten der Zustandsübergangsmatrix ausgeht.

6.4. *Modelliertes System*: (kurz)

Nicht-wiederherstellbar, wiederherstellbar, mit Behebung von transienten Fehlern; periodisch erneuerte, nicht-wiederherstellbare Systeme.

6.5. Modelleingabe

- Strukturierte Eingabeparameter:
  Initiale Anzahl von aktivierten Modulen und Ersatzmodulen, erlaubte Anzahl von Abschaltungen (*Degradations*) aus der Menge aktiver Module, Anzahl guter Module im Zustand des ersten sicheren Herunterfahrens, etc.

- Physikalische Eingabeparameter:
  Ausfallrate für jedes aktive Modul, für jedes Ersatzmodul, für jedes gute Modul im gesicherten heruntergefahrenen Zustand; Rate für das Auftauchen von transienten Fehlern für jedes aktivierte Modul und die mittlere Dauer eines transienten Fehlers.

- Logistische Eingabeparameter:

  – Anzahl der Reparatureinrichtungen; Reparaturrate für jedes Modul, falls das System sich nicht im komplett konfigurierten oder im heruntergefahrenen Zustand befindet sowie die Neustart-Rate, um aus dem Fehlzustand nach einem Crash wieder in den komplett-konfigurierten Zustand zu gelangen.

  – Erneuerungsprozess: Länge des Wartungsintervalls, Austausch- oder

Reparaturrate für jedes Modul, Neustartrate nach einem Crash-Fehlzustand, mittlere Abmeldedauer des Systems sowie das Gesamtzeit-Intervall des Beginns der i-ten Erneuerungsphase und der Wiederaufnahme des Systembetriebs in kompletter Konfiguration. Das Gesamtzeit-Intervall ist eine zufällige Variable, die mittels anderer Parameter geschätzt wird.

- Eingabeparameter zur Erkennung und Behandlung von permanenten Fehlern:
  Wahrscheinlichkeit der Behandlung (Erfassung) von Fehlern im Ersatzmodul, der Erfassung von aktivierten Modulen, der Erfassung von aktivierten Ersatzmodulen sowie der Erfassung des erfolgreich durchgeführten Herunterfahrens des Systems.

- Eingabeparameter zur Erkennung und Behandlung von transienten Fehlern:
  Anzahl der Wiederherstellungsphasen, Wiedererlangbarkeit (bedingte Wahrscheinlichkeit dafür, dass bei Auftreten eines Fehlzustandes dieser Zustand als nicht katastrophal einzustufen ist), Ausfallrate des zur Ausführung des transienten Wiederherstellungsprozesses verwendeten Hardware, Vektor der Dauer zur Systemwiederherstellung und der Effektivität der Systemwiederherstellung.

6.6. Modellausgabe

- Betriebsdauer-Maße für das System:
  Zuverlässigkeit, mittlere Zeit bis zum Auftreten des ersten Ausfalls, Ausfallrate, die Balance der Ausfälle jedes Teilsystems, aus der Redundanz resultierender Verbesserungsfaktor der Zuverlässigkeit, Verbesserungsfaktor der Betriebsdauer zwischen zwei konkurrierenden Designs.

- Lebenszyklus-Maße für das System:

  – Aggregierte Zustandswahrscheinlichkeiten:
    Wahrscheinlichkeit, dass das System betriebsbereit ist ohne Herabsetzungen in der Leistung oder der Fehlertoleranz; Wahrscheinlichkeit, dass das System betriebs-bereit ist in einem beliebigen eingeschränkten Zustand; Wahrscheinlichkeit, dass das System in einem unsicheren Zustand oder in

Zuständen mit einigen defekten, nicht behebbaren Ersatzmodulen betriebsbereit ist; Wahrscheinlichkeit, dass sich das System im gesicherten heruntergefahrenen Zustand befindet; Wahrscheinlichkeit eines katastrophalen Ausfalls.

– Stationäre Zustandswahrscheinlichkeiten zur Beschreibung des einschränkenden Verhaltens wiederherstellbarer und erneuerbarer Systeme:
Momentane Verfügbarkeit (Wahrscheinlichkeit, dass das System zur Zeit t betriebsbereit ist), Sicherheit (Wahrscheinlichkeit, dass sich das System in einem betriebsbereiten oder im sicheren heruntergefahrenen Zustand befindet), Verfügbarkeit bei einem möglichen spezifizierten Leistungsniveau (Kapazität), Durchschnitt erwarteter möglicher Leistungsniveaus (Kapazität), Häufigkeit von Ausfällen, mittlere verlorene Rechenzeit verursacht durch einen Crash, mittlere verlorene Rechenzeit verursacht durch sicheres Herunterfahren; mittlere Zeit, in der sich das System im heruntergefahrenen Zustand befindet; durchschnittliche Anzahl von Modulausfällen.

6.7. Schnittstellen

6.1.1. Eingabe

C- Implementierung (nicht verfügbar).

6.1.2. Ausgabe

C-Implementierung (nicht verfügbar).

7 Anwendungsfälle

Zur Unterstützung in der Lehre. Keine weiteren Anwendungsfälle bekannt.

8 Annahmen und Einschränkungen

Konstante Übergangsraten (Ausfallraten), Annahme unterschiedlicher Eigenwerte für die

Matrix der Übergangsraten, die zu einem Berechnungsaufwand der Lösung von $O(n^5)$ führt – statt einem Aufwand von $O(n^4)$, wie bei der klassischen Berechnung (nur bei Modellen wiederherstellbarer Systeme). Bei nicht-wiederherstellbaren (geschlossenen) Systemen mit mehrfachen Eigenwerten kann eine Lösung nur gefunden werden, falls garantiert werden kann, dass die Übergangsratenmatrix diagonalisierbar ist.

BQR CARE

1. *Quellen / Referenzen*

   BQR (2007): http://www.bqr.com

2. *Projektstatus*

   Dieses Tool ist ein kommerzielles Produkt der BQR Zuverlässigkeitstechnik. CARE gehört dabei zu einem größeren Sofwareprodukt, das die Zuverlässigkeit erhöhen und die Kosten der Entwicklung von elektromechanischen Bauelementen senken soll. Das Produkt der Firma, die seit 1989 existiert, besteht aus mehreren Modulen, die im Folgenden alle näher beschrieben werden.

3. *Lizenztyp*

   Auf der Internetseite von BQR sind keine kommerziellen Lizenzbedingungen aufgelistet. Das Tool CARE ist in Module aufgeteilt, von denen nur ausgewählte Teile erhältlich sind. Es ist möglich, eine preiswertere akademische Lizenz zu erhalten,

   obwohl der Preis nicht ausdrücklich erwähnt wird. Allerdings gibt es für die akademische Version Einschränkungen in der Anzahl der verwendbaren Elemente/Komponenten.

4. *Allgemeiner Zweck*

   Das Tool wurde für die Zuverlässigkeitsanalyse von technischen, elektronischen und

mechanischen Systemen (Mechatronik) entwickelt. Obwohl es für die Softwarezuverlässigkeit anwendbar ist, liegt der Fokus der Entwickler bei der Bereitstellung einer erweiterten Unterstützung für Zuverlässigkeitsschätzungen in der Mechatronik. So besteht das Tool aus Modulen, die eine durch Alterung, Spannungs-, Strom- und Temperaturschwankungen verursachte Veränderung in der Zuverlässigkeit modellieren und abschätzen. Ferner verwendet das Tool MTBF-Bibliotheken für elektronische Bauelemente basierend auf den häufig genutzten Zuverlässigkeitsstandards: HRD5, MIL-HDBK- 217 und Bellcore.

5. Plattformen

Windows 2K, Windows XP.

6. Modell

6.1. Modellklasse

Analytisch, Simulation, Mechatronik.

6.2. Modelltypen

Ausfallmodus, Auswirkungs- und Gefährdungsanalyse (FMEA/CA), Fehlerbaum-Analyse, Zuverlässigkeitsblockdiagramm. Für elektronische Komponenten werden verschiedene Zuverlässigkeitsstandards/-modelle genutzt (die für unseren Schwerpunkt allerdings nicht relevant sind): MIL-HDBK-217F Notice 2, 217Plus, British-Telecom HRD5, Siemens SN-29500, IEC62380 - RDF2000 / UTEC80810, NSWC98 mechanical, Bellcore Issue 6, CNET 2000, GJB299. Die Komponentenbelastungsanalyse wird basierend auf NAVSEA TE000-AB-GTP-010 Revision 2 ausgeführt. BIT und ATE führen die Testbarkeit durch (Analyse des Überdeckungsgrads der eingebauten Tests des Systems, Prüfung des Isolationsgrads der Tests).

6.3. Modellierte Systeme

Wiederherstellbare und nicht-wiederherstellbare Systeme.

6.4. Modelleingabe

Standardmöglichkeiten für FMEA/CA, FTA und RBD.

6.5. Modellausgabe

Für FMEA/CA: Ausfallmodus-Wahrscheinlichkeiten.
Für FTA: minimale Schnittmengen (mit Wahrscheinlichkeiten), Wahrscheinlichkeiten und Raten für alle Baumereignisse.
Für RBD: Zuverlässigkeit, Verfügbarkeit, Ausfallzeit, MTBCF, MTTR und Ausfallhäufigkeiten, Hazard-Rate, totale Ausfallzeit während der Aufgabe.
Für Reparaturbäume (beruhend auf der FTA Methodik): MTTR für alle zusammengefassten Baumknoten.

6.6. Schnittstellen

6.6.1. Eingabe

Über Benutzeroberfläche; direkte Eingabe von CAD/CAE Schnittstellen für MTBF-Analysen (Mentor Graphics, Cadence, Or-Cad, Excel CSV, ERP SAP, ERP MFG-Pro), Datenaustausch mit anderen BQR-Produkten.

6.6.2. Ausgabe

Graphisch; einige Module können Daten mit anderen BQR-Produkten austauschen.

7. Anwendungsfälle

BQR unterstützt Firmen in den folgenden Industriebereichen: Luft- und Raumfahrt, Automobil,
Chemie, Elektronik, Haushalts- und Unterhaltungselektronik, Verteidigung, Militär,

Energiewerke, Benzin und Öl, Medizin, öffentliche Verkehrsmittel und Halbleiter.
Die Hauptfirmen sind IBM, Siemens, Lockheed Martin, BAE, Fokker, Rafael, Elisra, Israel Aircraft Industries, Israel Electric Company, Tadian und Elta. Jedoch bietet BQR sowohl Tools
als auch Beratungsservices an. Deshalb ist es nicht ganz klar, ob die Tools direkt von den gelisteten Kunden, den Angestellten während einer Beratung oder gar nicht genutzt werden.

8. *Annahmen und Einschränkungen*

Das Tool verwaltet Hardware- bzw. Software-Ausfallbäume ohne Einschränkung der Funktionsebenen, der Anzahl der Bauteile oder der Anzahl der Ausfallarten. Ein FMECA-
Baum lässt sich in einen Fehlerbaum überführen (Voraussetzung: der Kauf des Fehlerbaum-Moduls). Die RBD Grundmodelle sind: seriell, parallel, K-aus-N und Stand-by mit oder ohne Reparatur. Netz- und Markov-Modelle gehören zu den fortgeschrittenen Techniken. Mit Hilfe der Markov-Erweiterung lassen sich fast alle komplexen Systemzustände simulieren, wie Auslöser für Transitionen (T*ransition Drivers*), Reparaturzeiten und verschiedene Unterblockzustände (*States of Sub Blocks*). Komplexe Netze, die möglicherweise Mehrfacheinträge und -austräge zum und vom System enthalten, können mit der Netzerweiterung simuliert werden. Innerhalb einer Netzeinstellung kann sich ein Unterblock (oder eine Verbindung) aus beliebigen Grund- oder Markov-Modellen zusammensetzen. Die folgenden Ausfallverteilungen können bei Grundblöcken eines RBD gewählt werden: Exponential-, Weibull-, Gleich-, Pareto- und Rayleigh-Verteilung sowie logarithmische Normal- und Badewannen-Verteilung. Mit Reparaturbäumen lassen sich MTTR-Berechnungen für jeden zusammengesetzten Block (Teilgraph oder Knoten) durchführen. MTTR wird aus dem Durchschnitt der Unterblockwartungszeiten (anhand der Ausfallhäufigkeiten) berechnet (Ersatz- und/oder Reparaturzeit sind abhängig vom Unterblock-Reparaturtyp). Die maximale Reparaturzeit eines Blocks lässt sich unter der Annahme einer logarithmischen Reparaturzeitverteilung bestimmen.

CARE III

1. *Quellen / Referenzen*

   Bavuso (1982)
   Bavuso (1984)
   Geist, Trivedi (1983)
   Stiffler et al. (1979)

2. *Projektstatus*

   Das Tool wurde von der NASA zusammen mit der Raytheon Company entwickelt als Nachfolger von CARE, um dessen Unzulänglichkeiten zu beseitigen. Seit der letzten Veröffentlichung von 1984 sind keine Artikel mehr erschienen, die sich mit diesem Tool beschäftigt haben. Der Autor (Bavuso) hat seine Erfahrung in die Entwicklung von HARP einfließen lassen.

3. *Lizenztyp*

   Unbekannt.

4. *Allgemeiner Zweck*

   Computer-Aided Reliability Estimation Program (CARE III) ist ein allgemeines Werkzeug zur Bestimmung der Zuverlässigkeit für sehr große, hochverfügbare Systeme.

5. *Plattform:*

   VAX-11

6. *Modell*

   6.1. Modellklasse

Analytisch, IT-Ebene.

6.2. Modelltyp

Unterstützte Modelle sind Fehlerbäume, nicht-homogene Markov-Ketten und Semi-Markov-Ketten.

6.3. *Modellbeschreibung*

Das Verhalten im Fehlerfall wird mit Fehlerbäumen spezifiziert und mit nicht-homogenen Markov-Ketten modelliert. Die Fehlerbehandlung wird mit Semi-Markov-Ketten modelliert. Unterstützt werden einfache und doppelte Fehler.

6.4. Modellierte Systeme

Hoch-zuverlässige, wiederherstellbare Systeme.

6.5. *Modelleingabe*

- Name der Stage (Bezeichner für ein Subsystem, welches Module mit gleicher Weibull-Verteilung für die Zeit bis zum Ausfall enthält),
- Anzahl initialer funktionsfähiger Module im Subsystem, Minimale Modulanzahl für eine betriebsbereite Stage, Modelle zur Fehlerbehandlung, Fehlertyp (permanent, transient oder intermittierend),
- Übergangsrate vom aktiven Fehlzustand in den gutartigen Fehlzustand (Auswirkungen des Fehlers sind verschwunden), Übergangsrate vom gutartigen Fehlzustand in den aktiven Fehlzustand, Rate der Erkennung von Fehlern durch einen Selbsttest, Rate der Fehlergenerierung durch einen Fehlzustand (*Error*), Rate der Fehlererkennung,
- Single-Point-Ausfallverteilung und Parameter für die vordefinierte Fehlerbaum-beschreibung, Anwender beschreibt Beziehung zwischen System-Stages mittels einer Fehlerbaumbeschreibungssprache. Beschreibungssprache unterstützt AND,

OR, K-aus-N, inverse Eingabegatter und bis zu 2000 globale Ereignisse und 70 Ereignisse mit Datumseingabe. Hardware und funktionale Redundanz sind beschrieben.

6.6. Modellausgabe

- Wahrscheinlichkeit, dass ein gegebenes „Modul in der Stage X“ (Modul(x)) nicht einen Fehler einer spezifizierten Kategorie zur Zeit t erfahren hat
- Zuverlässigkeit eines Moduls(x)
- Rate des Auftretens eines Fehlers einer spezifizierten Kategorie in einem gegebenen betriebsbereiten Modul(x), Rate des Auftretens eines Fehlers in den übrig gebliebenen, fehlerfreien Modulen zur Zeit t bei einer spezifizierten Anzahl von fehlerhaften Modulen summiert über alle Stages
- Wahrscheinlichkeit, dass ein gegebenes Modul(x) einen latenten Fehler einer spezifizierten Kategorie zur Zeit t hat, unter der Bedingung, dass es bereits in der Zeit t einige Fehlzustände erfahren hat
- Wahrscheinlichkeit, dass ein gegebenes Modul(x) einen latenten Fehler zur Zeit t hat, unter der Bedingung, dass es bereits in der Zeit t einige Fehlzustände erfahren hat
- Für eine gegebene Stage die Wahrscheinlichkeit, dass ein Subsystem eine spezifizierte Anzahl latenter Fehler enthält unter der Bedingung einer spezifizierten Anzahl an fehlerhaften Modulen
- Wahrscheinlichkeit, dass ein System mit einer spezifizierten gesammelten Anzahl an Fehlern eine spezifizierte gesammelte Anzahl an latenten Fehlern enthält
- Wahrscheinlichkeit, dass ein System mit einer spezifizierten Anzahl an Fehlern in einen superkritischen Zustand gerät, falls ein Fehler in einer gegebenen Kategorie zur Zeit t auftritt
- Wahrscheinlichkeit, dass ein System mit einer spezifizierten Anzahl an Fehlern in einen spezifizierten kritischen Zustand gerät, falls ein Fehler in einer gegebenen Kategorie zur Zeit t auftritt
- Wahrscheinlichkeit, dass ein Fehler einer gegebenen Kategorie zur Zeit t aktiviert wird unter der Bedingung, dass er zur Zeit t latent ist

- Wahrscheinlichkeit, dass ein System in einen spezifizierten kritischen Zustand zur Zeit t tritt und dieses Ereignis eventuell einen Systemausfall verursacht
- Wahrscheinlichkeit, dass das System mit einer spezifizierten Anzahl von Fehlern zur Zeit t sich in einem kritischen Zustand befindet
- Rate bei der Systeme mit einer spezifizierten Anzahl von Fehlern zur Zeit t ausfallen, in Folge von kritischen Fehlerbedingungen
- Wahrscheinlichkeit, dass sich das System von einem Fehler erholt

6.7. *Schnittstellen*

CARE III verfügt über eine interaktive Benutzerschnittstelle zur Modellierung des zu untersuchenden Systems.

7. *Anwendungsfälle*

Das Werkzeug wurde von der NASA zum Einsatz im Strahltriebwerkslabor entwickelt. Einsatzzweck war die Modellierung von Systemen im Avionikbereich, insbesondere von Computersystemen im Weltall.

8. *Annahmen und Einschränkungen*

Beschränkt auf hochverfügbare, nicht-wiederherstellbare Systeme. Der Abdeckungsgrad ist beschränkt durch die Schwierigkeit beim Lösen des Semi-Markov-Modells. Kann keine Ablaufmodelle darstellen. Kann keine „Cold Spares“ modellieren.

CARMS

1. *Quellen / Referenzen*

CARMS (2007): http://www.tc.umn.edu/~puk/carms.htm

2. *Projektstatus*

Das Projekt scheint nicht mehr weiterentwickelt zu werden. Die letzte Version ist von 2001 und für Windows 3.1.

3. *Lizenztyp*

   Frei verfügbar (keine Lizenzen auffindbar).

4. *Allgemeiner Zweck*

   CARMS (Computer-Aided Rate Modeling and Simulation) ist ein integriertes Markov-Modellierungs- und Simulationswerkzeug, um eine große Auswahl von zeitabhängigen, vorhersageorientierten Problemen zu lösen.

5. *Plattform:*

   Windows 3.1, 95, 2000, XP

6. Modell

   6.1. Modellklasse

   Analytisch, IT-Ebene.

   6.2. Modelltyp

   Markov-Ketten.

   6.3. Beschreibung

   CARMS ermöglicht die Modellierung mit einem graphischen Editor für Markov-Ketten und einer Tabellenkalkulation-ähnlichen Eingabemaske. Während der Simulation stellt das Programm ein interaktives graphischen Interface zur Verfügung. Zur Lösung einer

Reihe von verschiedenen Problemen stellt das Programm numerische Methoden zur Verfügung.

6.4. Modellierte Systeme

Es werden zeitabhängige, Vorhersage-orientierte Probleme modelliert.

6.5. Modelleingabe

Das Programm verarbeitet Markov-Ketten und benötigt deren Spezifikation (Zustände, Übergangswahrscheinlichkeiten), um das System zu analysieren.

6.6. Modellausgabe

Graphische Auswertung der Modellparameter, Kennzahlen und Berichte.

6.7. Schnittstellen

CARMS hat eine graphische Benutzerschnittstelle und bereitet Resultate graphisch auf. Diese können gespeichert und weiterverarbeitet werden.

7. Anwendungsfälle

Zuverlässigkeitsanalyse, Operation-Research, fehlertolerante Systeme, konstruktiver Entwurf sowie wissenschaftliche oder statistische Modellierung mittels Markov-Analyse.

Zuverlässigkeitsvorhersage, Design fehlertoleranter Systeme, Konstruktion von Zustands-diagrammen, Markov-Analyse, Markovsche Fehlerkettenmodellierung, Wartungs- und Verfügbarkeitsvorhersagen, Warteschlangen- und Bedienungsmodellierung, Systembestands-analyse, Wahrscheinlichkeitsevaluierung

8. Annahmen und Einschränkungen:

Nicht bekannt.

CASRE/SMERFS

1. *Quellen / Referenzen*

   CASRE (2007): http://www.openchannelfoundation.org/projects/CASRE_3.0
   Lyu, Schoenwaelder (1998)
   SMERFS (2007): http://www.slingcode.com/smerfs/

2. *Projektstatus*

   Die aktuelle Version stammt aus dem Jahre 2002, der momentane Stand des Projektes ist unbekannt. Es gibt weiterhin eine Web-basierte Version (Web-CASRE). Die Modellierung und Fähigkeiten der Modellanwendbarkeit stammen vom Public-Domain-Softwarepaket für Zuverlässigkeit SMERFS (*Statistical Modeling and Estimation of Reliability Functions for Software*).

3. *Lizenztyp*

   Lizenz der Open Channel Software, California Institute of Technology für private interne Zwecke.

4. *Allgemeiner Zweck*

   CASRE (*Computer-Aided Software Reliability Estimation*) ist ein Softwarewerkzeug zur automatischen Messung und Bewertung der Zuverlässigkeit von Software. Es beinhaltet die graphische Darstellung von Ausfalldaten, deren Vorverarbeitung (beispielsweise Filterung) sowie Ausfallprognosen für das entsprechende Softwaremodul.

5. *Plattform*

Die Originalversion ist Windows-basiert. Es gibt eine Web-basierte Version (Web-CASRE) für UNIX (Back-End, Client-Server-Architektur).

6. Modell

6.1. *Modellklasse:*

Quantitativ – analytisch, IT-Ebene.

6.2. Modelltyp

Jelinski-Moranda-Modell, nicht-homogenes Poisson-Prozessmodell.

6.3. Modellbeschreibung

Das Jelinski-Moranda-Modell wird zur Vorhersage der Zeit zwischen zwei Ausfällen (MTBF) verwendet, basierend auf den Zwischenzeiten von vergangenen Ausfällen.
Das nicht-homogene Poisson-Prozessmodell wird zur Messung der Fehleranzahl verwendet.

6.4. Modelliertes System

Wiederherstellbare Systeme (Software).

6.5. Modelleingabe

Ausfalldaten.

6.6. Modellausgabe

Fehleranzahl, Testintervall-Länge, Fehlerdichte, MTBF, Kumulative Anzahl von Ausfällen, Kumulative Anzahl von Ausfällen vom Beginn der Modellierung, Zuverlässigkeit.

6.7. Schnittstellen

Schnittstellen sind soweit nicht erkennbar, jedoch verwendet CASRE für die Modellierung die Bibliotheken von SMERFS, welches als Public-Domain-Software veröffentlicht wurde.

7. *Anwendungsfälle*

Softwaretest: CASRE kann als Werkzeug nach dem Unit-Test durchgängig zum System- und Akzeptanztest sowie für weitere Testoperationen verwendet werden.

8. *Annahmen und Einschränkungen*

Benutzbarkeit ist eingeschränkt durch die Begrenzung der Datei mit Fehlerdaten auf 64 KBytes (ca. 3000 Ausfalldaten pro Datei oder 2000 Testintervallen für die Zählung von Ausfalldaten). Die Anzahl der Modelle, die gleichzeitig gestartet werden können, hängt von der Größe des Hauptspeichers ab (CASRE simuliert alle Daten im Hauptspeicher).

CPNTOOLS

1. *Quellen / Referenzen*

CPNTOOLS (2007): http://wiki.daimi.au.dk/cpntools/cpntools.wiki
Wells (2006)

2. *Projektstatus*

CPNTOOLS werden an der Universität von Arhus, Dänemark, stetig weiterentwickelt. Die neueste öffentliche Version 2.2.0 ist von 2006.

3. *Lizenztyp*

Es existieren drei verschiedene Lizenztypen:

- kommerziell,
- Militär/Regierung und
- nicht-kommerziell/akademisch.

4. *Allgemeiner Zweck*

CPNTOOLS dienen zur Bearbeitung, Simulation und Analyse von Farbigen Petri-Netzen (*Coloured Petri Nets*). Mit Hilfe dieses Modells können insbesondere verteilte Prozesse modelliert werden, die miteinander kommunizieren und sich gegenseitig synchronisieren.

5. *Plattform:*

Windows 2000, Windows XP, Linux, Fedora Core 2
(OpenGL Hardware-Beschleunigung erforderlich).

6. *Modell*

6.1. Modellklasse

Analytisch, IT-Ebene.

6.2. *Modelltyp*

Farbige Petri-Netze, eine Erweiterung von Petri-Netzen.

6.3. *Modellbeschreibung:*

Farbige Petri-Netze sind eine graphisch-orientierte Sprache für Design, Spezifikation, Simulation und Verifikation von Systemen. Sie sind gut geeignet für Systeme, die aus einer Anzahl von Prozessen bestehen, die miteinander kommunizieren und sich

gegenseitig synchronisieren. Typische Beispiele von Anwendungsgebieten sind Kommunikations-protokolle, verteilte Systeme, automatisierte Produktionssysteme, Workflow-Analysen und VLSI-Chips.

6.4. *Modellierte Systeme*

Alle Systeme die sich mit Petri-Netzen darstellen lassen, insbesondere verteilte Systeme.

6.5. *Modelleingaben*

Das zu analysierende Petri-Netz.

6.6. *Modellausgaben*

Der Zustandsraum des Petri-Netzes kann analysiert werden und Eigenschaften wie Beschränktheit und Deadlock-Freiheit können bewiesen werden. Des Weiteren können Aussagen über die Leistung des Systems getroffen werden.

6.7. *Schnittstellen*

CPNTOOLS verfügen über eine leistungsfähige graphische Benutzeroberfläche zur Bearbeitung von Petri-Netzen.

7. *Anwendungsfälle*

Es existieren zahlreiche Anwendungsmöglichkeiten dieses Werkzeugs, unter anderem Verifikation von Netzwerkprotokollen, Analyse von Software, Workflow-Analysen und Geschäftsprozessmodellierung, Steuerungssysteme im Echtzeitbereich, militärische Planung und Flugverkehrskontrolle.

8. *Annahmen und Einschränkungen*:

Nicht bekannt.

DyQNtool+

1. *Quellen / Referenzen*

   Haverkort et al. (1992)
   Haverkort, Niemegeers (1996)

2. *Projektstatus*

   Das Werkzeug DyQNtool+ wurde von Haverkort und Maass an der Universität von Twente in den 90er Jahren entwickelt. Der aktuelle Status des Projektes ist unbekannt.

3. *Lizenztyp*

   Nicht bekannt (vermutlich existiert aber eine akademische Lizenz).

4. *Allgemeiner Zweck*

   DyQNtool+ verwendet das Konzept der dynamischen Warteschlangen-Netzwerke. Als solches basiert das Werkzeug auf einem Framework, in dem beide Seiten – Aspekte der Zuverlässigkeit als auch der Leistungsfähigkeit sowie ihrer gegenseitigen Abhängigkeit – formal spezifiziert werden können.

5. *Plattform*

   SUN-4-Systeme unter Unix. DyQNtool+ verwendet Pakete von SHARPE und SPNP.

6. *Modell*

   DyQNtool+ basiert auf Markovschen Reward-Modellen. Diese werden entlang der Richtlinien von Konzepten der dynamischen Warteschlangen-Netzwerke spezifiziert.

6.1. Modellklasse

Quantitativ – analytisch, IT-Ebene.

6.2. Modelltyp

Markovsche Reward-Modelle.

6.3. Modellbeschreibung

Aus den Markov-Ketten, welche von SPNP generiert und mittels verschiedener Leistungsanalysen von SHARPE bewertet wurden, lassen sich stationäre, transiente und kumulative Maße der Leistungsfähigkeit ableiten. DyQNtool+ generiert dazu die darunter liegende Markov-Kette und das SHARPE-Leistungsmodell. Es wird das Leistungsmodell innerhalb von SHARPE gelöst und mit den Zustandswahrscheinlichkeiten kombiniert, die aus SPNP abgeleitet werden.

6.4. Modelliertes System

Keine Angaben.

6.5. Modelleingabe

Als Modelleingabe dienen:

- Eine CSPL-Beschreibung des Zuverlässigkeitsmodells, die auch in SPNP direkt verwendet werden kann.
- Eine Beschreibung der parametrisierten Warteschlangen-Netzwerke in Form einer C-Implementierung.
- Eine Beschreibung der Abbildung der SPNP-Markierungen auf die Parameter

der Warteschlangen-Netzwerke in Form einer C-Implementierung.

- Leistungsmaße als Entlohnung in Form von C-Funktionen.

6.6. Modellausgabe

Zuverlässigkeit und Leistungsfähigkeit in Tabellenform.

6.7. Schnittstellen

C-Implementierung (nicht verfügbar).

7. *Anwendungsfälle:*

Nicht bekannt.

8. *Annahmen und Einschränkungen*

Eingeschränkt auf Warteschlangen-Netzwerke.

Eclipse TPTP (Test and Performance Tool Platform)

1. *Quellen / Referenzen*

ECLIPSE (2007): http://www.eclipse.org
ECLIPSE TPTP (2007): http://www.eclipse.org/TPTP

2. *Projektstatus*

Das Projekt ist derzeit aktuell und befindet sich unter ständiger Weiterentwicklung. Es gehört zu den Top-Level-Projekten innerhalb der Eclipse Foundation – einem inoffiziellen Zusammenschluss industrieller Softwareentwicklungsfirmen unter der Führung von IBM.

Die *Test-and-Performance-Tool-Plattform* (TPTP) ist die Weiterentwicklung des Eclipse-Hyades-Projektes. Die Weiterentwicklung basiert auf kollaborativer SW-Entwicklung und ist für jeden frei zugänglich.

3. *Lizenztyp*

Eclipse Open-Source-Lizenz.

4. *Allgemeiner Zweck*

TPTP liefert eine standardisierte, generische und erweiterbare Werkzeugplattform, mittels der Softwareentwickler eigene spezialisierte Werkzeuge erstellen können. Zusätzlich liefert TPTP auch Module, mit denen der Anwender bereits grundlegende Leistungs- und Testaufgaben bewältigen kann. Das TPTP-Projekt gliedert sich in die folgende Anwendungsbereiche (*Projects*):

- *Platform*: Liefert eine allgemeine Infrastruktur für Benutzerschnittstellen, Datenmodellen, Datensammlungen und Kommunikationssteuerungen. Darüber hinaus liefert die Plattform Ansatzpunkte (*Extension Points*) für Programmierer, um die Struktur um Funktionalität zu erweitern.
- *Testing Tools*: Liefert grundlegende Erweiterungen zur Durchführung von SW-Tests und Erstellung von Testumgebungen für Applikationen auf Basis des Plattformprojektes.
- *Tracing and Profiling Tools*: Erweitert das Plattformprojekt um Methoden zur Datensammlung (Heap- und Stack-Informationen) bei der Ausführung von einzelnen JAVA-Programmen oder verteilten Anwendungen, die das Common-Trace-Modell unterstützen. Zusätzlich wird eine graphische Darstellung Datenanalyse angeboten.
- *Monitoring Tools*: Erweitert das Plattformprojekt zur Sammlung und Analyse von Daten, die während der Ausführung entstehen (*Log*) sowie von statistischen Modellen. Diese Daten können auch in andere Formate oder Modelle umgewandelt werden, um Ursachen- und Musteranalyse zu betreiben. Typische Anwendungen sind CPU- und Speicherverbrauch.

5. *Plattform*

Diverse Plattformen (u.a. PC, SPARC) und Betriebssysteme (u.a. Windows, Linux) werden unterstützt.

6. *Modell*

6.1. *Modellklasse*:

Quantitativ - Analytisch, Benchmarking und Test, IT-Ebene.

6.2. *Modelltyp*

TPTP Monitoring Tools: Analyse und Vorverarbeitung (Filterung) von Logdateien.

6.3. *Modellbeschreibung*

Es werden mit Hilfe von Patterns und Matching-Regeln Aufzeichnungen aus dem Systemlauf (Logdateien) analysiert. Der Xpath Log-Analyzer benutzt Matching-Regeln in Form von Xpath-Ausdrücken.

6.4. *Modelliertes System*

Keine Angaben.

6.5. *Modelleingabe*

Logdateien oder Dateien aus Datenbankbeständen.

6.6. *Modellausgabe*

Es wird ein Vektor mit Objekten generiert, die Lösungsanweisungen für das Problem enthalten.

6.7. *Schnittstellen*

Aufgrund des Konzeptes der Plug-Ins sind zahlreiche Schnittstellen in Form von vordefinierten Schnittstellen (Extension Points) vorhanden.

7. *Anwendungsfälle*

Unterstützung der Testbeschreibungssprache TTCN-3 für Kommunkations-basierte Anwendungen (Client-Server-Applikationen).

8. *Annahmen und Einschränkungen*

Soweit bekannt ist TPTP auf JAVA-Anwendungen beschränkt. Geplant sind jedoch Erweiterung für Anwendungen anderer Programmiersprachen (vor allem in Verbindung mit dem C/C++ Development Toolkit).

ExhaustiF

1. *Quellen / Referenzen*

Exhaustif (2007): http://www.exhaustif.es

2. *Projektstatus*

Das Projekt ist aktuell, die Webseite wurde zuletzt 2007 aktualisiert. Das Werkzeug wird entwickelt aus einer Kooperation der Technischen Universität Madrid mit zwei kleineren spanischen Softwareunternehmen. Es gibt eine Variante des Programms als Plugin für IBM Rational Rose und für Eclipse. Es fehlen allerdings ausführliche Produktinformationen und Referenzen (Nachfragen per E-Mail blieben unbeantwortet).

3. *Lizenztyp*

Kommerzielle Lizenz.

4. *Allgemeiner Zweck*

ExhaustiF ist eine Werkzeug zur Ausführung von Black-Box- und Grey-Box-Tests basierend auf der SWIFI-Methode (*Software Implemented Fault Injection*). Mittels ExhaustiF lassen sich Design- und Programmfehler identifizieren. Das Werkzeug dient dazu, während des Entwicklungsprozesses (u.a. während des Integration- und Systemtests) Zuverlässigkeits- und Verfügbarkeitseigenschaften von softwareintensiven Systemen zu verbessern.

5. *Plattform*

RTEMS/ERC32 und RTEMS/Intel (*Real-Time Operating System for Multiprocessor Systems*). Es ist geplant ExhaustiF für Windows/Intel und Linux/Intel zur Verfügung zu stellen.

6. *Modell*

6.1. *Modellklasse*

Quantitativ – Benchmarking und Test, IT-Ebene.

6.2. *Modelltyp*

SWIFI (*Software Implemented Fault Injection*).

6.3. *Modellbeschreibung*

Das Werkzeug injiziert Fehler sowohl auf Programmebene (Prozeduren, Variablen) als auch auf Hardware-Ebene (CPU, Speicher, I/O). Das Testobjekt wird automatisiert in einer Host-Target-Umgebung ausgeführt.

6.4. Modelliertes System

Keine Angaben.

6.5. Modelleingabe

Programmquelltext. Parameter zur Injizierung von Fehlern auf Hardware-Ebene.

6.6. Modellausgabe

Verhalten des Testobjektes – vermutlich Metriken zur Leistungs- und Abdeckungsmaße. Nähere Informationen sind nicht verfügbar.

6.7. Schnittstellen

Die Eingabe erfolgt graphisch über eine Benutzeroberfläche. Die Ergebnisse werden im SQL-Datenbankformat abgespeichert.

7. *Anwendungsfälle*

Eine Beta-Testvariante soll bei EADS-Astrium verwendet werden.

8. *Annahmen und Einschränkungen*

Keine bekannt. Möglicherweise ist die Anwendung auf PC/Windows.

FAIL-FCI

1. *Quellen / Referenzen*

Horau et al. (2005)

Tixeuil et al. (2006)

2. *Projektstatus*

Das Projekt ist aktuell, die letzte Publikation stammt aus dem Jahre 2006. INRIA (*Institut national de recherche en informatique et en automatique*) ist ein Forschungsinstitut, dessen Schwerpunkte im Bereich Computerwissenschaft, Kontrolltheorie und Angewandte Mathematik liegen.

3. *Lizenztyp*

Es gibt keine Hinweise auf eine Veröffentlichung der Software.

4. *Allgemeiner Zweck*

FAIL-FCI (*Fault Injection Language - FAIL Cluster Implementation*) ist ein von INRIA France entwickeltes Werkzeug zur Zuverlässigkeitsbestimmung von Cluster- bzw. Grid-Systemen. FAIL-FCI erlaubt die Modellierung und Umsetzung von Fehlerszenarien mittels einer abstrakten Fehlerbeschreibungssprache und einem Verfahren zur Fehlerinjektion. Weiterhin kann die Effektivität fehlertoleranter Mechanismen von Cluster-Systemen durch FAIL-FCI getestet werden. Mittels Durchführung von Stresstests lassen sich weitere nicht-funktionale Eigenschaften wie Robustheit und Leistung durch FAIL-FCI quantifizieren.

5. *Plattform*

Linux.

6. *Modell*

6.1. Modellklasse:

Quantitativ – Benchmarking und Test, IT-Ebene.

6.2. Modelltyp

Software-basiertes Verfahren zur Fehlerinjektion.

6.3. Modellbeschreibung

Mittels FAIL werden vom Anwender Fehlerszenarien beschrieben. Diese werden vom FCI-Compiler zu C++-Quelltext kompiliert, mit FCI-Bibliotheken verlinkt und als archivierter Quellcode über das Grid zu den Knotenrechnern verteilt. Auf den Zielrechnern wird das Programm übersetzt und zur Ausführung gebracht (FCI-Dämon). Die verteilte Anwendung (*System Under Test*) wird vom FCI-Dämon zur Ausführung gebracht und entsprechend dem Fehlerszenario instrumentiert.

6.4. Modelliertes System

Verteilte Systeme (Grid).

6.5. Modelleingabe

Formatierte Beschreibung des Fehlerszenarios (FAIL-Sprache) in Textform.

6.6. Modellausgabe

C++-Quellcode mit Anweisungen zu fehlerhaften Verhalten der Ausführungsumgebung des Testobjektes (Software-basierte Fehlerinjektion) – beispielsweise probabilistisch erzeugte Prozessterminierung.

6.7. Schnittstellen

Keine bekannt.

7. Anwendungsfälle

Die Publikation enthält eine exemplarische Anwendungsbeschreibung zur Software-basierten Fehlerinjektion und dem Stresstest.

8. *Annahmen und Einschränkungen*

Keine bekannt.

FIGARO / KB3 Workbench

1. *Quellen / Referenzen*

Bouissou et al. (1991)
Bouissou (2002)
Bouissou, Lefebvre (2002)
Bouissou et al. (2002)
Bouissou, Bon (2003)
Bouissou, Muffat (2004)
Bouissou (2005)
Bouissou et al. (2005)
Breton et al. (2006)
Carer et al. (2003)
FIGARO (2007): http://www.edf.fr/72493m/txt/Home-fr/Research--Development/The-scientific-Community/Downloads/KB3.html
Gallois, Pillière (1991)

2. *Projektstatus*

FIGARO/KB3 wird entwickelt und verwendet von der EDF (*Electricite de France*). Das Werkzeug enthält eine Modellbeschreibungssprache (FIGARO), eine Benutzeroberfläche (*GUI*) und externe, austauschbare Modell-Lösungsmethoden (KB3 Workbench). All diese Bestandteile sind einzeln verfügbar.

3. *Lizenztyp*

   Eine kommerzielle Lizenz wird durch Apsys vergeben (http://www.apsys.eads.net/). Zudem wird eine zeitlich unbegrenzte Testlizenz angeboten. Die einzige Beschränkung besteht hierbei
   in der Größe der Studien, die verarbeitet werden können. Diese dürfen nie mehr als 80 Objekte
   umfassen.

4. *Allgemeiner Zweck*

   Der allgemeine Zweck dieses Werkzeugs ist die Berechnung der Systemzuverlässigkeit, der Verfügbarkeit und der Leistung mittels vieler unterschiedlicher Modelle. Die Besonderheit dieser Lösung liegt in der Verwendung von Wissensdatenbanken zur Modellierung abstrakter
   Systeme. Verschiedene Übersetzer leiten daraus automatisch die Daten ab, welche für das klassische Zuverlässigkeitsmodell von Bedeutung sind (z.B. für Markov-Ketten und Petri-Netze).

5. *Plattform*

   Unix/Xwindow, MS Windows.

6. *Modell*

   6.1. Modellklasse:

   Analytisch, Simulation, IT-Ebene.

   6.2. *Modelltyp*:

   Markov-Ketten, Boolean logic Driven Markov Process (BDMP), Fehlerbäume, Zuverlässigkeitsblockdiagramme, Petri-Netze

6.3. Modellbeschreibung:

Das System wird mittels der abstrakten Modellierungssprache FIGARO beschrieben, die sich für objektorientierte Modelle eignet. Diese Beschreibung wird vom System durch eine vordefinierte Wissensdatenbank in ein geeignetes Modell überführt, beispielsweise in ein Petri-Netz oder ein Zuverlässigkeitsblockdiagramm. Durch Hinzufügen verschiedener Werkzeuge zur Modellanalyse kann eine Lösung des Modells unter Verwendung der Monte-Carlo-Simulation, der Sequenzgenerierung und -quantifizierung sowie der Markov-Ketten-Generierung und -Quantifizierung (analytisch) bestimmt werden.

6.4. Modellierte Systeme

Wiederherstellbare und nicht-wiederherstellbare Systeme.

6.5. Modelleingabe

Die generischen Bestandteile (Objektklassen), Knoten (Quellen oder Terminale) und Kanten (Vertices), werden eingegeben. Zusätzlich werden die Eigenschaften jeder Klasse definiert: Ausfallmodelle und ihre Parameter, das Auftreten von Ausfällen und Reparaturen sowie Abhängigkeiten. Nach dem Aufbau eines solchen Modells werden Klassen als Objekte instantiiert, die das System repräsentieren. Zu diesem Zweck wird eine graphische Benutzeroberfläche bereitgestellt. Danach überführt das System das Eingabemodell in ein lauffähiges Modell unter Verwendung einer geeigneten Wissensdatenbank. KB3 ist auch mit einem AR2FIG0-Konverter ausgestattet. Dieser akzeptiert Beschreibungen in der Datenfluss-Sprache AltaRica und überführt diese nach FIGARO. Dadurch ist es möglich, sich die Rechenleistung der Werkzeuge FIGSEQ, YAMS und FIGMAT-SF zu Nutze zu machen und die existierenden Modelle in KB3 zu verwenden.

6.6. Modellausgabe

Zuverlässigkeit und kurzzeitige Zuverlässigkeit (Markov-Ketten) Zuverlässigkeit, Verfügbarkeit, durchschnittliche Leistung, Anzahl der Ereignisse, Verweildauer in Zustandsklassen (Simulationen), Zuverlässigkeit von wiederherstellbaren und nicht-wiederherstellbaren Systemen (Sequenzgenerierung und -quantifizierung).

6.7. Schnittstellen

6.7.1. Eingabe

FIGARO stellt eine objektorientierte Sprache zur Definition der Modelleingabe und eine graphische Benutzeroberfläche zur Erzeugung von Modellbeschreibungen zur Verfügung. Dabei werden ASCII-Dateien als Ausgabe generiert, die dann als Eingabe für Analyse-Werkzeuge verwendet werden können. Weiterhin kann mit der Schnittstelle auch eine Modellprüfung durchgeführt werden.

6.7.2. Ausgabe

KB3 bietet verschiedene Formen der Ausgabe: Rohdaten als Textdateien oder automatische visuelle Aufbereitung (Graphen, Bilder).

7. Anwendungsfälle

Verfügbarkeitsstudien werden zur Optimierung des Designs oder der Ausnutzung gewerblicher Systeme durchgeführt, z.B. bei Produktionseinrichtungen und öffentlichen Netze (wie Elektrizitäts-, Telefon- und Verkehrsnetze). Weiterhin wird das Werkzeug zur Zuverlässig-keitsabschätzung von Hightech-Systemen (On-Board-Systeme, medizinische Geräte) und zur Sicherheitskontrolle gefährlicher Industrieprozesse verwendet, z.B. in der Atomwirtschaft und der Chemieindustrie.

8. Annahmen und Einschränkungen

Die Sequenzgenerierung durch Ableitungen kann nur für den Faktor Zuverlässigkeit Resultate liefern. Falls die Größe des Systems ansteigt (Markov), wächst die Anzahl der

Zustände exponentiell. Zudem ist die Generierung von Konfidenzintervallen bei der Wahrscheinlichkeit von seltenen Ereignissen sehr aufwendig, die nicht mit Markov modelliert wurden.

GRAMP/GRAMS

1. *Quellen / Referenzen*

   Dolny et al. (1983)
   Dolny, Fleming (1983)
   Fleming (1980)
   Fleming, Dolny (1984)

2. *Projektstatus*

   Die Entwicklung von GRAMS begann als Thema einer Doktorarbeit (PhD) Fleming (1980) und wurde später in Fleming, Dolny (1984) fortgesetzt, basierend auf den Werkzeugen ARIES, SURF und CARE III. Der momentane Projektstatus ist unbekannt.

3. *Lizenztyp*

   Die letzte bekannte Implementierung stammt aus dem Jahre 1984. Der Lizenztyp ist allerdings
   unbekannt.

4. *Allgemeiner Zweck*

   Das GRAMP-Werkzeug liefert eine Empfindlichkeitsanalyse und wird zur Modellierung folgender Aspekte fehlertoleranter Systeme benutzt: Fehleraufdeckung, vorbeugende Wartung, Anschaffungskosten, Betriebskosten und Support-Kosten. Zudem wird es zur Vorhersage der Zuverlässigkeit, Wartbarkeit und Kosten des Lebenszyklus eines Systems verwendet, welches von GRAMP modelliert wurde.

5. Plattform

FORTRAN-77-Implementierung für VMS.

6. Modell

6.1. Modellklasse:

Simulation, IT-Ebene.

6.2. Modelltyp:

Zeitkontinuierliche Markov-Modelle (GRAMP), gelöst mittels der diskreten Monte-Carlo-Simulation (GRAMS).

6.3. *Beschreibung*:

GRAMP und GRAMS bieten während der Konzept- und Designphase eine Methodik zur Verifizierung der Zuverlässigkeit, Wartbarkeit und Kostenspezifizierung von fehlertoleranten Systemen an. Diese Methode berücksichtigt systematisch und quantitativ das Design, die Wartbarkeit und den Support innerhalb der Analyse durch Berechnung von Unsicherheiten unter Verwendung der Sensitivitätsanalyse des Front-Ends. Das System wird mittels zeitkontinuierlicher Markov-Modelle modelliert und simulativ gelöst. Zeitlich sich verändernde Ausfallraten werden als stückweise, konstante Ausfallraten gehandhabt.

6.4. Modelliertes System:

Nicht-wiederherstellbare und wiederherstellbare Systeme.

6.5. Modelleingabe:

GRAMP

- System:
  Festgelegte Aufwendung für Reparaturen, Ausfallkosten für den Zusammenbruch des Systems, Laufoptionen, Druckoptionen, Dauer des Einsatzes, Beschaffungskosten (System- und Modulebene), durchschnittliche Reparaturkosten.

- Teilsystem:
  Anzahl der Module im Teilsystem, Anforderungen zur Zuverlässigkeit, Option zur opportunistischen Wartung (Falls ein Modul repariert ist, erlaubt dieser Parameter dem Benutzer zu entscheiden, ob die Wartungsstrategie die Suche nach weiteren fehlerhaften Modulen beinhaltet, damit diese ebenfalls repariert werden können.), Struktur des Teilsystems (kritische Mengen).

- Module:
  Grad der vorbeugenden Wartung, der Redundanz und der Ersetzung, zu berechnende Empfindlichkeit, Ausfallraten, Abdeckungsaktivierte und Ersatz-Einheiten (*Coverage-active and Spare Modules*)

GRAMS

- System: Mit verschiedenen Wartungsarbeiten verbundene Kosten:

  - Kosten der erforderlichen Wartung eines Backup-Systems, wenn es bei der Vorabinspektion versagt; angefallene Kosten, falls der Umschaltmechanismus für das Backup-System nach dem Versagen des Primärsystems nicht funktioniert; Kosten für die opportunistische Wartung an einer oder mehreren Komponenten, die wiederhergestellt werden mussten, während sich der Hauptcomputer entweder in der Erreichung seiner maximalen Betriebsdauer MOT (*Maximum Operating Time*) oder in planmäßiger Wartung befand
  - Kosten, die aus der planmäßigen oder außerplanmäßigen Wartung von Line-Replacable-Units (LRU) entstehen
  - Kosten, die in Verbindung mit einem Austausch des Hauptcomputers (Host),

seiner Versendung zur Abteilung und der Reparatur bei planmäßiger oder außerplanmäßiger Wartung stehen.

- Ereigniswahrscheinlichkeiten, die für die Eingabe auf Systemebene benötigt werden:
  - Wahrscheinlichkeit des Einsatzes nach Typ A
  - Wahrscheinlichkeit des Beenden eines Einsatzes, obwohl nicht erforderlich
  - Wahrscheinlichkeit für das Ausbleiben einer Wartung nach Beenden eines Einsatzes, obwohl erforderlich
  - Wahrscheinlichkeit einzelner Ereignisse (beispielsweise Maximaltemperatur und Blitzschlag)
  - Wahrscheinlichkeit für das Eintreten des Schweregrades n für diese Ereignisse
  - Wahrscheinlichkeit des Ausfalls des Hauptcomputers während eines Einsatzes
  - Wahrscheinlichkeit von Ausfällen während der Inspektion des Backup-Systems
  - Wahrscheinlichkeit von Ausfällen beim Umschalten von und zum Backup-System. Anzahl der Betriebsstunden des Hauptcomputers pro Lebenszeit und pro Jahr, Länge des Einsatzes A und B in Stunden, Anzahl der Schweregrade n für einzelne Ereignisse, Anzahl der zu simulierenden Hauptcomputer, Anzahl aller Hauptcomputer, Zeitreihe für die Daten der Ausfallrate, Hauptcomputer-Population und die minimale und maximale Anzahl von Hauptcomputern

6.6. Modellausgabe

GRAMP

- Kostenevaluierungsmodell CEM (*Cost Evaluator Model*):

  - Ausfall- und Wartungsereignisse pro Millionen Stunden, mittlere Zeitspanne zwischen den Ereignissen, relative Operations- und Supportkosten (O & S), Beschaffungskosten, Gewicht,

Empfindlichkeit

- Zuverlässigkeit-Durchführbarkeits-Modell (auf Modul- oder Teilsystembasis): Zuverlässigkeit, mittlere Zeit bis zum Ausfall, Graph der Zuverlässigkeit in der Vergangenheit

- Systembuchführung (Systembasis): Zuverlässigkeit, MTTF (*Mean Time To Failure*), O & S- und Anschaffungskosten, Gesamtbesitzkosten, beinhaltet Treibstoff, Laufzeit, unterbrechungsfreie Wartung, Kosten der Reparatureinrichtung, Gewicht, CEM Ereignisse pro Million Stunden (Abbruchrate), MTBF (*Mean Time Between Failures*), MTBR (*Mean Time Between Repair*), Zusammenfassung der oberen sieben Kategorien für jedes Teilsystem

GRAMS

- Eingabe der Namensliste, Parameterdefinitionen, Teilsystem- und Moduldefinitionen, Systemergebnisse
- Zuverlässigkeit der Teilsysteme:
  - Teilsystem-MTBF, Modul-MTBF bezogen auf Komponentenausfall / bei Ausfall der Ausfallabdeckung (*Failure Coverage*)
  - System-MTBR in Abhängigkeit vom Wartungstyp, MTBR durch Teilsystemwartung (Mittel über die Lebenszeit des Hauptcomputers)
  - MTBR durch jährliche Teilsystemwartung, Ereignisse hervorgerufen durch Teilsystemwartung (jährliches Mittel über die Lebenszeit des Hauptcomputers)
  - Ereignisse hervorgerufen durch jährliche Teilsystemwartung
- O & S-Kosten bei einem Zusammenbruch des Systems, Austausch von Komponenten abhängig vom Wartungstyp (jährlicher Durchschnitt über die Lebenszeit), Austausch von Komponenten abhängig vom Wartungstyp (jährlich), Austausch von Komponenten nach dem Wartungsplan (jährlich und gesammelt), Ausfälle des Backup-Systems, Definition der Ausgabeoptionen (es wird festgelegt, welche Ausgaben gedruckt werden sollen)

6.7. Schnittstellen

FORTRAN-77-Implementierung (nicht verfügbar).

7. Anwendungsfälle

GRAMP und GRAMS wurden in verschiedenen Phasen von Projekten zum Design von fehlertoleranten elektronischen Controllern angewendet. Sie wurden dazu benutzt, zu entscheiden, ob die Verbesserungen in Zuverlässigkeit, Betriebssicherheit und Funktion (beispielsweise Verringerung des Stromverbrauchs) die Gesamtkosten der Produktion eines elektronischen Controllers begünstigten. Zusätzlich wurde eine Sensitivitätsanalyse durchgeführt, um über Unwägbarkeiten in der Abdeckung, unähnlicher Redundanz, Hardware-Redundanz und Aufrechterhaltung der Zuverlässigkeit innerhalb der Kosten zu entscheiden. Während der finalen Phase der Designauswahl wurden GRAMP und GRAMS als Analysewerkzeuge implementiert, um Konfigurationen des Controllers zu entwerfen, die die geforderten Spezifikationen der Zuverlässigkeit und Betriebssicherheit erfüllen. Zuverlässigkeits- und Wartbarkeitszielstellungen wurden ebenfalls auf Basis von konstruktiven Einschränkungen ausgewertet:

- Das Ziel des FAFTEEC-Programms war es, einen Designansatz für ein autonomes (*Full-Authority*) elektronisches Steuerungssystem zu entwickeln unter dem vorrangigen Gesichtspunkt der Zuverlässigkeit. Der Ansatz, um dieses Ziel anzugehen, war die Festlegung einer Basisversion eines autonomen elektronischen Steuerungssystems für ein hoch entwickeltes Kampfflugzeug und waren anschließende Verbesserungen dieser grundlegenden Steuerung in Bezug auf bestimmte Zielstellungen mittels Redundanz, Wiederherstellungsstrategien und Wartbarkeitsphilosophien einzuarbeiten.

  Designkandidaten für das Steuerungssystem wurden zusätzlich zu ihrer Fähigkeit, die Designzielstellungen zu erfüllen, in Bezug auf Kosten und Gewicht evaluiert. Das grundlegende Steuerungssystem wurde modularisiert,

um beschreibbare Komponenten (Pumpen, Thermoelemente, Aktuatoren, etc.) zu erhalten. Für diese Komponenten wurden Informationen zur Zuverlässigkeit und den Kosten gesammelt. Viele dieser Konfigurationen wurden mittels GRAMP überprüft. GRAMS testete vielversprechende Konfigurationen von GRAMP mittels eines zeitabhängigen Analyseansatzes basierend auf Monte-Carlo-Techniken. Die Ergebnisse der Analyse unter Verwendung von GRAMP und GRAMS zeigten die notwendigen Kosten- und Gewichtssteigerungen, wodurch Verbesserungen in der Einsatzzuverlässigkeit von einer Größenordnung im Vergleich zum Basissystem erreicht wurden.

Weitere Informationen:
Full-Authority Fault-Tolerant Electronic Engine Control (FAFTEEC) Program sponsored by the Air Force Wright Aeronautical Laboratories/POTC Dolny, Fleming (1983).

- Phase der endgültigen Designauswahl für einen elektronischen Controller für einen kommerziellen Auftraggeber: GRAMP und GRAMS wurden zur Verifizierung des Systems anhand von Zielstellungen zur Zuverlässigkeit angewendet. Spezifikationen zur Zuverlässigkeit wurden angegeben in Form von Anzahl von Ausfällen pro Millionen Stunden und die maximale Wahrscheinlichkeit von Ausfällen in einer beliebigen Stunde während eines 100-Stunden-Einsatzes Fleming, Dolny (1984).

8. Annahmen und Einschränkungen

Das Werkzeug kann maximal 1500 Markov-Zustände verarbeiten. Der Einsatzschwerpunkt liegt eher bei militärischen als bei kommerziellen Systemen.

- Systemannahmen: Zufällige Verschlechterung (Ausfall als stochastischer Prozess), unabhängige Komponentenausfälle, (stückweise) konstante Ausfallraten, Zustandsübergänge folgen augenblicklich Komponentenausfällen, Komponenten sind entweder funktionstüchtig oder

ausgefallen.

- Wartungsannahmen:
  Taktik der stationäre Wartung, Komponente wird nach einer Reparatur als neu angesehen, Wartungsarbeiten erfolgen sofort nach Komponentenausfall, sofortige Reparatur, unbegrenzte Betriebskapazität. Falls ein Teilsystem ausfällt, wird alles wiederhergestellt; Einschränkungen der Wartungstaktik gehorchen kohärenten Systemwiederherstellungsmodellen.

- Zeitannahmen: Kontinuierliche Zeit, ein Ausfall pro Zustandsübergang.

HARP

1. *Quellen / Referenzen*

   Bavuso et al. (1985)
   Bavuso et al. (1994)
   Dugan et al. (1985)
   Geist, Trivedi (1983)

2. *Projektstatus*

   HARP (*Hybrid Automated Reliability Predictor*) wurde von 1986-1994 kontinuierlich bis zur Version 7.0 an der Duke University weiterentwickelt. Seitdem sind aber keine neuen Versionen bekannt geworden.

3. *Lizenztyp*

   Unbekannt.

4. *Allgemeiner Zweck*

HARP ist ein allgemeines Tool zur Vorhersage von Zuverlässigkeit und Verfügbarkeit mittels Markov-Ketten. Als Eingabe dient ein Fehlerbaum oder graphische Eingabe durch den Benutzer.

5. *Plattform*

HARP läuft auf MS/PC-DOS, Microsoft Windows NT, OS/2, DEC VMS und Ultrix, Berkeley UNIX 4.3 und AT&T UNIX 6.2.

6. *Modell*

6.1. Modellklasse

Analytisch, IT-Ebene.

6.2. Modelltyp

Folgende Modelltypen werden unterstützt:

- FORM (*Fault Occurence and Repair Model*):
  Enthält Informationen über die Struktur der Hardware, die Prozesse über das Auftreten von Fehlen und manuelle (*offline*) Reparaturen. Spezifiziert entweder als Fehlerbaum oder Markov-Kette.

- FEHM (Fault/Error Handling Model):
  Beschreibt permanente, sporadische und transiente Fehler und Modelle zur Beschreibung von automatischen Reparaturen (*online*).

- Markov-Ketten

6.3. Modellbeschreibung

HARP wird benutzt, um mit einem hybriden Modell Aussagen über Zuverlässigkeit und

Verfügbarkeit zu machen. Als Modellierungsformat dient eine Textdatei, die über die graphische Benutzeroberfläche erzeugt werden kann. Darin werden das FORM- und das FEHM-Modell spezifiziert. Aus diesen Informationen wird eine Markov-Kette generiert und anschließend durch das Werkzeug gelöst. Die 16-Bit-Version kann bis zu 500 Zustände abbilden, die 32-Bit-Version bis zu 10 000 Zustände.

6.4. Modellierte Systeme

Wiederherstellbare und nicht-wiederherstellbare Systeme.

6.5. Modelleingaben

- Feherbaumrepräsentation in graphischer Form, Markov-Modellrepräsentation in graphischer Form, Ausfallraten mit Variationsbandbreite (Variation Band), Reparaturrate, Initialbedingungen, beinahe zufällige Fehler abhängig vom Ort, beinahe zufällige Fehler abhängig von der Komponente, beinahe zufällige Fehler ignorieren

- Eingaben für das Standard ESPN-Modell zur Fehlerbehandlung:

  - ACTIVE-Übergang (von aktiviert zu gutartig intermittierend)
  - BENIGN-Übergang (von gutartig zu aktiviert intermittierend)
  - Lebenszeit eines transienten Fehlers
  - DETECT-Übergang (von DETECT zu Counter)
  - ERROR-Übergang (von ERROR zu „Erkannt“ oder Systemausfall)
  - ERROR-DETECT-Übergang (nur von ERROR zu DETECT)
  - ISOLATION-Übergang (von permanenter Fehler zu Rekonfiguration oder Systemausfall)
  - RECOVERY-Übergang (von transienter zu vollständiger Wiederherstellung)
  - RECONFIGURATION-Übergang (von Rekonfiguration zu eingeschränktem Betrieb oder zum Systemausfall)
  - Wahrscheinlichkeit der Fehlererkennung bei einem Selbsttest
  - Wahrscheinlichkeit der Fehlererkennung

- Wahrscheinlichkeit der Isolierung erkannter Fehler
- Anzahl der Wiederherstellungsversuche
- Wahrscheinlichkeit erfolgreicher Rekonfigurierung
- Anteil transienter Fehler
- gewünschtes S-Konfidenzniveau
- zulässige Fehler

• Beschreibung der Art des Wiederherstellungsprozesses in Petri-Netz-Terminologie.

6.6. Modellausgaben

• Empfindlichkeit der Zuverlässigkeits- oder Verfügbarkeitsvorhersagen, um Variationen und initiale Zustandsunsicherheiten zu parametrisieren. Diese Information liefert ein Maß der Systemempfindlichkeit für Designfehler. Zuverlässigkeit (Unzuverlässigkeit) und momentane Verfügbarkeit (Nichtverfügbarkeit) in Abhängigkeit von der Zeit.

• Ausfallwahrscheinlichkeiten hervorgerufen durch Ermüdung redundanter Elemente, Single-Point-Ausfall, beinahe-zufälliger Fehler, Wahrscheinlichkeit für beinahe-zufällige Fehler.

6.7. Schnittstellen

HARP hat eine graphische Benutzeroberfläche, kann aber auch Modelle von CARE III und ARIES verarbeiten. Modellspeicherung und -weitergabe kann über Textdateien geschehen.

7. Anwendungsfälle

Keine Angaben.

8. Annahmen und Einschränkungen

HARP ermöglicht keine Modellierung von Wartungsstrategien und kann keine Modelle mit zeitlichen Abhängigkeiten oder „Cold Spares“ generieren.

Isograph

Isograph ist eine integrierte Lösung zur Zuverlässigkeitsprüfung und -verbesserung. Das Isograph-Paket besteht aus gesonderten Tools, welche die Abschätzung von Zuverlässigkeit, Wartbarkeit, Verfügbarkeit, Lebensdauer oder Betriebssicherheit ermöglichen. Die im Folgenden beschiebenen Werkzeuge sind im Reliability Workbench Program enthalten.

2.2.1.1 FaultTree+

1. *Quellen / Referenzen*

   FaultTree+ (2007a): http://www.isograph-software.com/ftpover.htm
   FaultTree+ (2007b): http://www.isograph-software.com/_techspecs/psa32techspec.pdf
   Trivedi (2003)

2. *Projektstatus*

   FaultTree+ ist ein kommerzielles Werkzeug, das kontinuierlich weiterentwickelt wird. Die letzte Vollversion V11 stammt aus dem Jahre 2005. Seit 2007 ist die zuletzt veröffentlichte Version 11.1 verfügbar.

3. *Lizenztyp*

   Informationen zur kommerziellen Lizenz und zum Preis wurden nicht bekannt gegeben, eine Evaluierungsversion ist erhältlich.

4. *Allgemeiner Zweck*

   Das Werkzeug bietet eine Zuverlässigkeitsanalyse, so dass Fehler- und

Ereignisbaumanalysen durchgeführt werden können. Spezifizierte Markov-Modelle können mit Ereignissen im Fehler- oder Ereignisbaumdiagramm verlinkt werden. Es ist allerdings auch möglich, voneinander unabhängige Analysen zu realisieren.

5. *Plattform*

Microsoft Windows.

6. *Modell*

6.1. Modellklasse

Analytisch, IT-Ebene.

6.2. Modelltypen

Fehler- und Ereignisbäume, Markov-Modelle.

6.3. Modellbeschreibung

Die Fehlerbaummethode umfasst die Erzeugung von Fehlerbaumdiagrammen aus Gattern
und Elementarereignissen, die die logische Beschreibung eines Systemausfalls (TOP-Ereignis) bezüglich des Ausfalls der Komponenten repräsentieren. Nachdem das Diagramm erstellt wurde, lassen sich Ausfallcharakteristiken des Systems darstellen. Zur Vervollständigung des Modells wird eine Systemanalyse durchgeführt. Dafür bestimmt FaultTree+ vorerst die minimalste Komponentenmenge, die einen Systemausfall hervorrufen kann (minimale Schnittmenge). Zum Abschluss berechnet FaultTree+ die quantitativen Parameter wie Nichtverfügbarkeit und Ausfallhäufigkeit des Systems. FaultTree+ beinhaltet die Möglichkeit zur Ereignisbaumanalyse. Das Ereignisbaummodell lässt sich unabhängig von dem Fehlerbaummodell erzeugen. Es kann aber auch die Ergebnisse der Fehlerbaumanalyse als Eingabe der Ereignisbaumwahrscheinlichkeiten verwenden. Zudem ist es möglich Markov-Modelle,

die sich unabhängig von der Fehlerbaumanalyse untersuchen lassen, als Quelle elementarer Ereignisdaten zu konstruieren.

6.4. Modellierte Systeme

Wiederherstellbare und nicht-wiederherstellbare Systeme.

6.5. Modelleingabe

Fehlerbäume:

- Gatter und Ereignisse
- Gatterparameter: Typ (OR, AND, VOTE, NOT, XOR, INHIBIT, PRIORITY, TRANSFER oder NULL), Name und Beschreibung, vorangegangene Ergebnisse, Gatter- und Ereigniseingaben, Anmerkungen, Hyperlinks zu externen Dokumenten
- Ereignisparameter: Ausfallmodell (RATE, FIXED, MTTF, DORMANT, SEQUENTIAL, ET INITIATOR, STANDBY, TIME AT RISK, BINOMIAL, POISSON, RATE/MTTR, WEIBULL, FIXED-PHASED, RATE-PHASED), Name, Beschreibung, CCF-Modell (*Common Cause Failure*), Logikmodus, Ausfallbefehle, allgemeines Ausfallmodell, Anmerkungen, Hyperlinks

Ereignisbäume:

- Das Ereignisbaummodell lässt sich unabhängig vom Fehlerbaummodell erstellen. Es
  kann aber auch die Gatterergebnisse der Fehlerbaumanalyse als Eingabe der

  Ereignisbaumwahrscheinlichkeiten nutzen.

Markov-Modelle:

- Markov-Modelle sind diskrete Systemzustände und -übergänge, die zeitabhängig sind. Die Modelle des Markov-Analyse-Moduls können zu den Elementarereignissen der Fehler- und Ereignisbaumanalyse-Module verlinkt werden.

6.6. Modellausgabe

Fehlerbäume:
CCF-Analyse, Bedeutsamkeitsanalyse (Fussel-Vesley, Birnbaum, Barlow-Proschan und sequentielle Bedeutsamkeitsmessungen), Unsicherheitsanalyse (Konfidenzintervalle können
durch unsichere Ereignisausfälle und unsichere Reparaturdaten geprägt sein) und Sensitivitätsanalysen (ermöglicht automatische Veränderung der Ereignisausfall- oder Reparaturdaten innerhalb vorgegebener Schranken), zeitabhängige Analysen (bestimmt dazwischen liegende Werte zeitabhängiger Systemparameter), Fehlerbaum-House-Event- Analyse, vollständige oder minimale Schnittmenge, Berechnungen der oberen Schranke und Risikoanalysen

Ereignisbäume:
Risikoberechnung, vollständige oder minimale Schnittmengenanalyse, Risiko-Bedeutsamkeitsanalyse (Erkennung der Hauptrisikofaktoren), Sensitivitätsanalysen (ermöglicht automatische Veränderung der Ereignisausfall- und Reparaturdaten innerhalb vorgegebener Schranken)

Markov-Modelle:
Nichtverfügbarkeit, Verfügbarkeit, Unzuverlässigkeit, Zuverlässigkeit, Ausfallhäufigkeit (unbedingte Ausfallintensität), Reparaturhäufigkeit (unbedingte Reparaturintensität), bedingte Ausfallintensität, bedingte Reparaturintensität, Anzahl erwarteter Ausfälle, Anzahl erwarteter Reparaturen; bezüglich der Lebensdauer: durchschnittliche Nichtverfügbarkeit, durchschnittliche Verfügbarkeit, erwartete Gesamtausfallzeit, erwartete Gesamtlaufzeit

6.7. Schnittstellen

6.7.1. Eingabe:

Alle drei Modellierungsmodi (Fehler- und Ereignisbäume sowie Markov-Modelle)

verwenden eine Windows-basierte graphische Benutzeroberfläche, die die Modellkonstruktion, -bearbeitung und -verwaltung ermöglicht.

6.7.2. Ausgabe:

Die gesamten Isograph Werkzeuge verwenden einen integrierten Reportgenerator als Ausgabeschnittstelle. Dieser wird in einem späteren Abschnitt gesondert betrachtet.

7. *Anwendungsfälle*

Luft- und Raumfahrt, Verteidigung, Automobil-, Eisenbahn-, Chemie-, Benzin- und Ölindustrie sowie medizintechnik.

8. *Annahmen und Einschränkungen*

Nicht bekannt.

2.2.1.2 Avsim+

1. *Quellen / Referenzen*

AvSim+ (2007a): http://www.isograph-software.com/avsover.htm
AvSim+ (2007b): http://www.isograph-software.com/_techspecs/avsim32techspec.pdf

2. *Projektstatus*

Avsim+ wird stetig weiterentwickelt, die zuletzt veröffentlichte Version V10 stammt aus dem Jahre 2006.

3. *Lizenztyp*

Die Testversion ist schon jetzt erhältlich. Informationen zum Preis der kommerziellen Lizenz wurden bisher nicht bekannt gegeben.

4. *Allgemeiner Zweck*

Verfügbarkeits- und Zuverlässigkeitssimulation komplexer, abhängiger Systeme.

5. *Plattformen*

Microsoft Windows.

6. *Modell*

6.1. Modellklasse

Simulation, IT-Ebene.

6.2. Modelltypen

Fehlerbaum, Netzwerk-(Zuverlässigkeitsblock-)Diagramm

6.3. Modellbeschreibung

Die logischen Wechselwirkungen von Ausfällen sind mittels Fehlerbäumen und Netzwerk- Diagrammen modelliert. Diese Diagramme werden zur Modellierung der Ausfälle und der Funktionsfähigkeit eines Systems oder der Durchsatzraten im System verwendet. Dabei können die Auswirkungen jeder beliebigen Ebene der logischen Diagramme zugeordnet werden, um die Auswirkungen von Ausfällen aufzuzeigen (wirtschaftlich, betrieblich, sicherheitstechnisch oder ökologisch). Arbeits-, Sicherungs- und Ausfalldaten können zusammen mit Informationen jeder funktionsfähigen Phase und Anweisungen für die Arbeitsgruppen importiert werden. Das Werkzeug analysiert das System unter Verwendung der Monte-Carlo-Simulation, um Verfügbarkeits- und Zuverlässigkeitsparameter, Kosten der Lebenszyklen und Bedeutsamkeitsranglisten zu erstellen. Zusätzlich lassen sich Wartungsintervalle optimieren.

6.4. Modellierte Systeme

Wiederherstellbare und nicht-wiederherstellbare Systeme.

6.5. Modelleingabe

Netzwerkelemente, Fehlerbaumelemente, Ausfallarten, Wartungsmodelle, Datenmengen (Zeit bis zu einem Ausfall), Konsequenzen, Ersatzteile, Arbeitsgebiete, Ausstattung, Arbeitsgruppen und Phasen.

6.6. Modellausgabe

Vergangenheitsdatenanalyse (*Historical Data Analysis*):
Avsim+ ermöglicht eine Weibull-Analyse von Daten vergangener Ausfälle und Wiederherstellungen. Diese ist durch Zuordnung der Wahrscheinlichkeitsverteilungen möglich, die die Ausfall- und Wiederherstellungscharakteristiken eines vorgegebenen Ausfallmodus (*Failure Mode*) darstellen. Die Ausfallverteilung, die einer gegebenen Datenmenge (Menge der Zeiten bis zu einem Ausfall) zugeordnet ist, kann zu Ausfallmodellen gehören, die in Zusammenhang mit den Blöcken von Netzwerkdiagrammen oder Fehlerbäumen stehen. Das Weibull-Modul analysiert dann die Zeiten bis zu einem Ausfall und einer Wiederherstellung unter Verwendung von exponentiellen eins-, zwei- und drei-parametrisierten Verteilungen (Weibull-, Bi- und Tri-Weibull-Verteilung), logarithmischen Normal- und Weibayes-Verteilung sowie schrittweisen Bi-, und Tri-Weibull-Verteilungen. Zudem passt es die ausgewählten Verteilungen an die Daten automatisch an und stellt die Ergebnisse graphisch in Form von kumulativen Wahrscheinlichkeitsdiagrammen sowie unbedingten Wahrscheinlichkeits-verteilungs- und bedingten Wahrscheinlichkeitsverteilungsdiagrammen dar.

Reserveoptimierung:
Mit AvSim+ lassen sich die Auswirkungen verschiedener Reserveelemente auf die Lebenskosten simulieren. Das Programm führt Simulationsdurchläufe für jede Kombination von Reserveelementen durch. Nachdem alle Simulationsdurchläufe

absolviert wurden, gibt AvSim+ die optimale Kombination der Reserveelemente für die Kosten der Lagerung an.

Wartungsoptimierung:
Um zu bestimmen, ob es sinnvoll wäre, die geplante Wartung oder Komponentenprüfung durchzuführen, kann AvSim+ verwendet werden. Es wird der optimale Wartungszeitraum für die geplanten Wartungsarbeiten und Arbeitsbetrachtungen ermittelt. Dies erfolgt durch ständige Veränderung des Wartungsintervalls und wiederholter Simulation der Lebenskosten.

Simulationüberwachung:
Die Simulationsüberwachung wurde für die Logikprüfung des Systemverfügbarkeitsmodells entwickelt. Während einer Simulationsüberwachung führt das Programm die Simulation schrittweise aus. Es stoppt immer dann, wenn eine Ereignisveränderung aufgetreten ist. Damit kann der Status bei jedem Schritt abgerufen werden.

Lebenskosten und Produtionskapazität:
Die Kosten und Produktionskapazitäten lassen sich abgesehen von Verfügbarkeit und Zuverlässigkeit modellieren. Dabei werden Arbeits-, Sicherungs- und andere Kosten in Betracht gezogen. Die Auswirkungen werden den Systemausfällen zugeordnet, wodurch die Ausfallkosten in die Berechnung einbezogen werden.

Sicherheits-, Ökologie- und Funktionseinfluss:
Durch Zuordnung von Sicherheits-, Ökologie- und Funktionskonsequenzen zu ausgewählten Systemausfällen können Häufigkeit und Dauer jeder dieser Konsequenzen bestimmt werden. Es lassen sich Härtegrade ermitteln, so dass die Faktoren Sicherheit, Ökologie und Funktionalität über die gesamte Lebensdauer des Systems bestimmt werden können.

6.7. Schnittstellen

6.7.1. Eingabe

Durch AvSim+ ist es möglich Fehlerbäume und Netzwerkdiagramme (Zuverlässigkeitsblockdiagramme) per Drag-and-Drop zu erstellen. Bei Fehlerbäumen wird das Diagramm automatisch aus logischen Verknüpfungen konstruiert. Für Zuverlässigkeitsblockdiagramme werden die Blöcke platziert und logische Verknüpfungen generiert. Damit leitet das Programm automatisch die Ausfalllogik des Systems ab. Mit einer Wizard-basierten graphischen Benutzeroberfläche werden die Elementeigenschaften festgelegt.

6.7.2. Ausgabe

Die gesamten Isograph-Werkzeuge verwenden einen integrierten Report Generator als Ausgabeschnittstelle. Dieser wird in einem späteren Abschnitt gesondert beschrieben.

7. *Anwendungsfälle*

Luft- und Raumfahrt, Verteidigung, Automobil-, Eisenbahn-, Chemie-, Benzin- und Ölindustrie sowie medizintechnik.

8. *Annahmen und Einschränkungen*

Nicht bekannt.

2.2.1.3 Reliability Workbench

1. *Quellen / Referenzen*

Reliability Workbench (2007a): http://www.isograph-software.com/_techspecs/avsim32techspec.pdf
Reliability Workbench (2007b): http://www.isograph-software.com/techspecs/wk32techspec.pdf

2. *Projektstatus*

Dieses Werkzeug wird kontinuierlich weiterentwickelt, die zuletzt veröffentlichte Version V10.1 stammt aus dem Jahre 2007.

3. Lizenztyp

Informationen zum Preis der kommerziellen Lizenz wurden bisher nicht bekannt gegeben. Eine Testversion ist verfügbar.

4. *Allgemeiner Zweck*

Reliability Workbench ist eine integrierte Umgebung, mit der Zuverlässigkeits- und Wartungsvorhersagen, FMECA (*Failure Mode, Effect and Criticality Analysis*), Zuverlässigkeitsblockdiagramm-, Fehlerbaum-, Ereignisbaum- und Markov-Analysen sowie Zuverlässigkeitszuordnungen gemacht werden können. Da Zuverlässigkeitsblockdiagramme, Fehlerbäume und Markov-Analysen bereits beschrieben wurden (Reliability Workbench bietet für diese Werkzeuge eine vereinheitlichte graphische Benutzeroberfläche an), wird hier der Fokus auf FMECA, Zuverlässigkeits- und Wartungsvorhersagen liegen.

5. *Plattform*

Microsoft Windows.

6. *Modell*

6.1. Modellklasse

Verschiedene Vorhersagemodelle, hierarchische Blockdiagramme.

6.2. Modelltyp

Analytisch, IT-Ebene.

6.3. Modellbeschreibung

Die Methoden zur Ausfallvorhersagen definieren ein System und die Bedingungen, unter denen es arbeitet (z.B. die Temperatur). Anschließend wird durch den Vorhersagealgorithmus die Berechnung der Ausfallhäufigkeit nach dem festgelegten Standard ausgeführt und das Ergebnis zurückgegeben. Die Reliability Workbench unterstützt die folgenden Ausfallvorhersagemodelle (Standards): MIL-HDBK-217, Telcordia SR-332, NSWC-98, IEC TR 62380 und GJB/Z 299B. Es können aber weitere Modelle hinzugefügt werden. Eine FMECA wird zur Erkennung eventueller Ausfallarten innerhalb eines Systems und ihrer Klassifizierung entsprechend ihres Härtegrads verwendet. Gewöhnlich wird diese Analyse in zwei Schritten ausgeführt. Im ersten werden Ausfallarten und ihre Aus-wirkungen (FMEA) ermittelt, während im zweiten Schritt diese Ausfallarten entsprechend der Kombination von Härte und Eintrittswahrscheinlichkeit eingeordnet werden (Kritikalitätsanalyse).

6.4. Modellierte Systeme

Wiederherstellbare und nicht-wiederherstellbare Systeme.

6.5. Modelleingabe

Zuverlässigkeitszuordnung:
Während der Entwurfsphase eines Produktes ist es häufig erforderlich, die Systemzuverlässigkeit zu bestimmen. Damit stellt sich die Frage, wie das Zuverlässigkeits-ziel des Systems erreicht werden kann. Mit einer Zuverlässigkeitszuordnung lassen sich die Zuverlässigkeitsziele individueller Teilsysteme bestimmen. Die einfachste Methode hierfür ist es, die Zuverlässigkeitsziele gleichmäßig unter allen Teilsystemen aufzuteilen. Jedoch ergibt sie selten die beste Verteilung von Zuverlässigkeitszielen für alle Teilsysteme. Besser ist es, Zuverlässigkeitswerte auf den Teilsystemen zu verteilen, die auf Komplexität, Kritikalität, erreichbare Zuverlässigkeit und anderen geeigneten Faktoren basieren.

Das Zuordnungsmodul der Reliability Workbench stellt sechs Methoden zur Bestimmung
der Zuverlässigkeitswerte des Teilsystems bereit:

1. Nicht beschränkte gleichmäßige Zuordnung
2. Nicht beschränkte gestaffelte Zuordnung
3. Nicht beschränkte proportionale Zuordnung
4. Nicht beschränkte redundant proportionale Zuordnung
5. Nicht beschränkte Zuverlässigkeitsneuzuordnung
6. Beschränkte direkte Erfahrungszuordnung

Innerhalb des Moduls kann eine Systemhierarchie erstellt werden, wo Teilsysteme bis zur
Komponentenebene aufgesplittet werden können.

Ausfallvorhersage:

Die Komponenten, die ein System ausmachen, können als Baumstruktur definiert werden. Der Baum besteht komplett aus Einzelteilen oder ist in Blöcke unterteilt, wovon jeder sich wiederum aus Blöcken oder Einzelteilen zusammensetzt. So werden Systeme und Teilsysteme auf eine einfache Weise abgebildet. Das Ausfallhäufigkeitsmodell für diese Komponenten setzt sich aus einer Grundausfallhäufigkeit für diesen spezifischen Komponententyp und der Vervielfachung von Faktoren, die als Pi-Faktoren bekannt sind, zusammen. Diese Faktoren hängen von den Betriebsbedingungen ab, denen die Komponenten ausgesetzt sind.

FMECA

- Definition des zu analysierenden Systems
- Konstruktion eines hierarchischen Blockdiagramms
- Erkennung von Ausfallarten auf allen Vertragsebenen
- Zuordnung der Auswirkungen zu den Ausfallarten
- Zuordnung der Härteklasse zu den Auswirkungen
- andere Daten der Ausfallarten, wie Ausfallerkennungsmethoden und

-häufigkeiten

- Klassifizierung der Ausfallarten nach Härte und Kritikalität

Ein großer Teil der eingegebenen Daten bei der Durchführung einer FMECA ist beschreibender Text. Das FMECA-Modul stellt eine Basisausdruck-Bibliothek bereit, die häufig verwendete Beschreibungen der Komponententeile, der Ausfallarten und -auswirkungen beinhaltet. Diese Ausdrücke können in Beschreibungsfelder eingetragen werden, indem der benötigte Ausdruck aus der Bibliothek ausgewählt wird, wodurch der Tippaufwand vermindert und die Kontiniuität gesichert wird. Es ist möglich eigene Ausdrucksbibliotheken zu erstellen oder die Grundausdrucksbibliothek zu ergänzen.

6.6. Modellausgabe

Bei der Ausfallvorhersage:
Vorhersage der durchschnittlichen Reparaturdauer, der Temperaturauswirkungen, der Belastungs- und Umgebungsveränderungen des Systems oder der Ausfallhäufigkeit, Wartungs-, Teilsystem- oder Geräteausfallhäufigkeitsvorhersage.

Bei der FMECA:
Ausfallauswirkungen, Härtegrade und Ausfallgründe durch die Systemhierarchie.

6.7. Schnittstellen

6.7.1. Eingabe

Ausfallvorhersage:
Die graphische Benutzeroberfläche ermöglicht die Auswahl des Ausfallvorhersage-modells und die dynamische Eingabe relevanter Parameter.

FMECA:
Die graphische Benutzeroberfläche ist mit Gestaltungs-, Eigenschafts-, Vererbungs- und Kapselungsmöglichkeiten zur Konstruktion eines hierarchischen Blockdiagramms ausgestattet.

6.7.2. Ausgabe

Die gesamten Isograph-Werkzeuge verwenden einen integrierten Reportgenerator als Ausgabeschnittstelle. Dieser wird in einem späteren Abschnitt gesondert beschrieben.

7. *Anwendungsfälle*

Luft- und Raumfahrt, Verteidigung, Automobil-, Kern-, Eisenbahn-, Chemie-, Benzin- und Ölindustrie sowie medizintechnik

8. *Annahmen und Einschränkungen*

Nicht bekannt.

2.2.1.4 NAP (Network Availability Program)

1. *Quellen / Referenzen*

NAP (2007a): http://www.isograph-software.com/napover.htm
NAP (2007b): http://www.isograph-software.com/_techspecs/nap32techspec.pdf

2. *Projektstatus*

Die zuletzt veröffentlichte Version V1.0 stammt aus dem Jahre 2005. Seitdem sind keine weiteren Versionen erschienen.

3. *Lizenztyp*

Informationen zum Preis der kommerziellen Lizenz wurden bisher nicht bekannt gegeben. Eine Evaluierungsversion ist verfügbar.

4. Allgemeiner Zweck

   NAP ermöglicht die Vorhersage von Verfügbarkeit und Zuverlässigkeit von Kommunikations-netzwerken.

5. *Plattformen*

   Microsoft Windows.

6. *Modell*

   6.1. Modellklasse

   Analytisch, IT-Ebene.

   6.2. Modelltyp

   Erweitertes Zuverlässigkeitsblockdiagramm.

   6.3. Modellbeschreibung

   Das Verfügbarkeitsmodell eines NAP-Netzwerkes verwendet eine erweiterte Zuverlässigkeitsblockdiagramm-Methodik, die die typischen Eigenschaften von Netzwerkelementen und ihrer Verbindungen behandelt. Neben der Vorhersage der Netzwerkverfügbarkeit stellt NAP auch eine Rangliste der Kritikalität bereit, die die schwachen Netzwerkstellen erkennen lässt.

   6.4. Modellierte Systeme

   Elemente von Kommunikationsnetzwerken.

   6.5. Modelleingabe

NAP erkennt die Ausfalllogik der Netzwerke anhand des eingegebenen Netzwerkdiagramms. Das Diagramm macht deutlich, wie individuelle Netzwerkelementausfälle mit anderen Netzwerkelementausfällen zusammenwirken, um einen Datenfluss zwischen Ausgangs- und Zielknoten im Netzwerk zu verhindern. Ein eingegebenes Netzwerkdiagramm legt die möglichen Kommunikationspfade zwischen verschiedenen Netzwerkelementen fest. Die Pfade zwischen Netzwerkelementen, die durch Verbindungen bestimmt sind, können gerichtet oder ungerichtet sein. Bei einer gerichteten Verbindung können die Daten nur in eine Richtung fließen. Das Netzwerkdiagramm enthält Blocksymbole, die gewöhnlich die Netzwerkelemente darstellen, und Leitungen, die die
Elemente miteinander verknüpfen. Die Parameter für jedes Netzwerkelement sind der Typ, das Ausfalldatenformat, die Ausfallhäufigkeit, das vorgegebene MTTR und die Pfadlogik.

6.6. Modellausgabe

Nichtverfügbarkeit des Netzwerkes, Verfügbarkeit, Ausfallhäufigkeit, MTTF, MTTR, MTBF, Unzuverlässigkeit, Zuverlässigkeit und gesamte Ausfallzeit.

6.7. Schnittstellen

6.7.1. Eingabe

Mit der graphischen Benutzeroberfläche ist es möglich, ein erweitertes Zuverlässigkeits-blockdiagramm zu erstellen. Die NAP-Bibliothekseinrichtung erlaubt die Konstruktion einer eigenen Bibliothek von Bauelement- und alternativen Bauelementlisten mit hierarchischer Struktur. Zudem können dann Bauelemente per Drag-and-Drop in ein Netzwerkelement oder Netzwerkblockdiagramm bewegt werden. Die hierarchische Struktur der Bibliothek ermöglicht das einfache Lokalisieren von Bauelementen anhand des Händlers, des Bauelementtyps und jeder anderen Kategorisierung. Projekte können auch an einen zentralen Teil der Bibliothek angehängt werden, um optional automatische Updates von Teildaten zuzulassen. Alternative Bauelementlisten können ebenfalls der Bibliothek zugefügt

werden, um eine schnelle Auswahl geeigneter Bauelementtypen innerhalb eines Netzwerkelementes zu ermöglichen.

6.7.2. Ausgabe

Die gesamten Isograph-Werkzeuge verwenden einen integrierten Reportgenerator als Ausgabeschnittstelle. Dieser wird in einem späteren Abschnitt gesondert beschrieben.

7. *Anwendungsfälle*

Kommunikationsnetzwerke.

8. *Annahmen und Einschränkungen*

Nicht bekannt.

2.2.1.5 AttackTree+

1. *Quellen / Referenzen*
AttackTree+ (2007a): http://www.isograph-software.com/atpover.htm
AttackTree+ (2007b): http://www.isograph-software.com/_techspecs/attacktree%2BV1TS.pdf

2. *Projektstatus*

Die aktuelle Version ist V1.0, allerdings ist das Erscheinungsdatum nicht bekannt.

3. *Lizenztyp*

Informationen zum Preis der kommerziellen Lizenz wurden bisher nicht bekannt gegeben. Eine Evaluierungsversion ist verfügbar.

4. *Allgemeiner Zweck*

Ein Attack-Tree ermöglicht die präzise Modellierung von Gefährdungen der Systemsicherheit in einem graphischen Format. Unter Verwendung des Attack-Tree-Modells kann die Wirkungsweise der Internet-, Netzwerk-, Bankensystem-, Einrichtungs- und Personalsicherheit modelliert werden.

5. *Plattform*

Microsoft Windows.

6. *Modell*

6.1. Modellklasse

Analytisch, Simulation, Service-Ebene.

6.2. Modelltyp

Attack-Tree.

6.3. Modellbeschreibung

Durch die Verwendung von Attack-Tree-Modellen ist es mit AttackTree+ möglich, die Wahrscheinlichkeit für den Erfolg verschiedener Angriffe zu modellieren. Zudem können mit diesem Werkzeug Indikatoren definiert werden, die die Kosten eines Angriffs, die operationale Schwierigkeit, einen Angriff zu erstellen und jedes andere relevante, quantifizierbare Maßsystem, das von Bedeutung sein könnte, quantifizieren können. Fragen, wie „Welche Angriffe haben die höchste Erfolgswahrscheinlichkeit zu geringen Kosten des Angreifers?“ oder „Welche Angriffe haben die höchste Erfolgswahrscheinlichkeiten ohne benötigte Spezialausrüstung?“, können mit diesem Werkzeug beantwortet werden. Mit AttackTree+ können verschiedene

Auswirkungsgkategorien und -stufen den Knoten eines Attack-Trees zugeordnet werden. Ein erfolgreicher Angriff kann wirtschaftliche, politische und betriebliche Auswirkungen haben. Im Unterschied zu einem gänzlich erfolgreichen Angriff kann ein nur zum Teil erfolgreicher Angriff Auswirkungen auf einer anderen Stufe haben.

6.4. Modellierte Systeme

Angriffsgefährdete IT-Systeme und Services.

6.5. *Modelleingabe*

Zu Beginn des Verfahrens, das zu der Konstruktion eines Attack-Trees für das Zielsystem führt, werden die möglichen Angriffsstellen identifiziert. Da verschiedene Angriffe ähnliche Methoden verwenden können, kann dies zu verketteten Attack-Trees führen. In der nächsten Phase werden alle möglichen Angriffsmethoden identifiziert, die an den erreichten Stellen ausgeführt werden können. Diese ersten zwei Phasen werden die höchste Ebene des Systemmodells erzeugen. Jeder Angriff setzt sich eventuell aus zahlreichen Bedingungen zusammen, die erfüllt werden müssen, um einen Angriff erfolgreich werden zu lassen oder er besteht einfach nur aus einem quantifizierbaren Ereignis. Folglich ist der nächste Schritt in der Attack-Tree-Konstruktion, jeden Angriff in seine Hauptbedingungen zu zerlegen. Dies führt zu einer vollständigen Attack-Tree-Struktur, bei der jede Abzweigung in einem einzelnen quantifizierbaren Ereignis endet. Wenn die Struktur komplettiert wurde, ist es erforderlich, die jeweilige Angriffshäufigkeit anzugeben. Danach sollte jede Ereigniswahrscheinlichkeit festgelegt werden, d.h., wie wahrscheinlich es ist, dass ein Angriff aus jeder Hinsicht Erfolg hat. Zudem können Attack-Trees Indikatoren einsetzen, um die Kosten des Angreifers aufzuzeigen, ob beispielsweise irgendeine Spezialausrüstung benötigt wird. Zu jedem Ereignis wird ein Wert für jeden Indikatortyp zugeordnet. Letzendlich ermöglicht AttackTree+ die Angabe von Konsequenzen und das Anfügen dieser an einen beliebigen Zweig im Attack-Tree. So lassen sich die Konsequenzen eines erfolgreichen Angriffs auf das Zielsystem modellieren. Dieses Feature ist besonders nützlich, wenn es viele TOP-Ereignisse gibt, die unterschiedliche Angriffstypen repräsentieren, oder Zweige im Attack-Tree vorhanden sind, die einen partiellen Erfolg eines Angriffs darstellen.

6.6. Modellausgabe

- Minimale Schnittmengen für jeden Zweig
- Wahrscheinlichkeit aller Schnittmengen
  - Indikatorwerte für jede Schnittmenge
  - Wahrscheinlichkeit aller Zweige
  - Indikatorwerte für jeden Zweig
  - Minimale Schnittmengen für alle Auswirkungen
  - Wahrscheinlichkeit aller Auswirkungen
  - Risikowerte aller Auswirkungsklassen

Dieser Vorgang ermöglicht es alle Kombinationen von Ereignissen geordnet nach den Erfolgswahrscheinlichkeiten zu betrachten, die zu einem erfolgreichen Angriff führen werden. Die entstehende Liste kann entsprechend der Indikatorwerte gefiltert werden (z.B. Schnittmengen für geringe Kosten des Angreifers). Es ist möglich, den Attack-Tree zu „beschneiden“, um den einfachsten Pfad zu ermitteln, bei dem der Angriff erfolgreich sein wird. Zudem können Gewichtungsranglisten betrachtet werden, um zu bestimmen, wie die Betriebssicherheit am effizientesten verbessert werden kann.

6.7. Schnittstellen

6.7.1. Eingabe

Mit Hilfe der graphischen Benutzeroberfläche ist es möglich, Attack-Trees zu konstruieren und zu beschreiben.

6.7.2. Ausgabe

Die gesamten Isograph-Werkzeuge verwenden einen integrierten Reportgenerator als Ausgabeschnittstelle. Dieser wird in einem späteren Abschnitt gesondert beschrieben.

7. *Anwendungsfälle*

   Internetanwendungen, Bankwesen und Netzwerke.

8. *Annahmen und Einschränkungen*

   Nicht bekannt.

2.2.1.6 Isograph Report Generator

Alle Isograph Werkzeuge verwenden den Isograph Report Generator als Ausgabeschnittstelle, welcher Ausgabeformate, wie Texte, Graphen und Diagramme unterstützt. Die Textbeschreibung wird festgelegt durch eine Auswahl der Teile der Projektdatenbank, die in Form von SQL-Ausdrücken im Dokument enthalten sein sollen. Die graphischen Berichte enthalten Diagramme mit Achsen, Beschriftungen, Legenden und Titeln, die durch einfache Dialoge konfiguriert werden. Solche Graphen können zwei- oder dreidimensional sein und ihre Position ist definierbar. Bei der Verwendung von Diagrammberichten können unter anderem Zuverlässigkeitsnetzwerke und Fehlerbäume in das Ausgabedokument eingefügt werden. Diese Dokumente können in verschiedenen Datenformaten vorliegen, wie RTF (bei Verwendung von Graphen und Diagrammen) oder als CSV-Dateien (nur bei Texten). Die letzte Variante ermöglicht eine flexible Weiterverarbeitung.

MARK

1. *Quellen / Referenzen*

   Babcock et al. (1987)
   Lala (1983a)
   Lala (1983b)
   Sharma, Bazovsky (1993)
   Smith, Lala (1986)

2. *Projektstatus*

Die erste Version dieses Werkzeugs wurde in Lala (1983a) und Lala (1983b) veröffentlicht. Die letzte bekannte Veröffentlichung wurde in Sharma, Bazovsky (1993) beschrieben .

3. *Lizenztyp*

Die letzte bekannte Implementierung stammt aus dem Jahre 1986. Der Lizenztyp ist unbekannt.

4. *Allgemeiner Zweck*

Auswertung der Zuverlässigkeit von komplexen Systemen, deren Eigenschaften mittels Markov-Ketten modelliert werden können. Das Werkzeug berechnet Zustands-wahrscheinlichkeiten als Funktionen über die Zeit, MTBF sowie die Wahrscheinlichkeit der durchschnittlichen Verweildauer in einem Zustand (*Average State Occupancy*).

5. *Plattform*

PL/1.

6. *Modell*

6.1. Modellklasse

Analytisch, IT-Ebene.

6.2. Modelltyp

Zustandsdiskrete, zeitkontinuierliche Markov-Modelle.

6.3. Beschreibung

Nicht verfügbar.

6.4. Modellierte Systeme

Jedes nicht-wiederherstellbare System, dessen Eigenschaften mittels Markov-Ketten modelliert werden können.

6.5. Modelleingabe

Spezifizierung des Modells: Anzahl von Zuständen im Modell, Beschreibung jedes Zustandes, Verweildauer für jeden Zustand bei gleicher Startzeit, Übergangsraten zwischen den Zuständen (definiert die Abhängigkeit zwischen Zuständen); Zeitspanne, innerhalb der das Markov-Modell zu lösen ist; Befehle, um Ergebnisse aus den Modellen zu mischen, um Systemzustandswahrscheinlichkeiten zu erhalten.

6.6. Modellausgabe

Graphen verschiedener Zustandswahrscheinlichkeiten als Funktionen über die Zeit (zum Beispiel: Graph der Wahrscheinlichkeit für ein gegebenes Modell, dass sich das System in einem gegebenen Zustand befindet oder sich nicht darin befindet); Graph der MTBF, Graph von durchschnittlichen Zustandsverweil-Wahrscheinlichkeiten.

6.7. Schnittstellen

Nicht verfügbar.

7. Anwendungsfälle

Zuverlässigkeitsvorhersage für Fehlertolerante Multiprozessoren (FTMP) , entwickelt für die NASA (Smith, Lala (1986)). Flugzeugsysteme und Betriebsicherheitssysteme in Kernkraftwerken (Sharma, Bazovsky (1993)).

8. Annahmen und Einschränkungen

Verwendet lediglich Exponentialverteilungen, Abdeckung kann nicht spezifiziert werden, kann nicht für wiederherstellbare Systeme verwendet werden und kann nicht mit transienten und intermittierenden Fehlern umgehen.

METFAC

1. *Quellen / Referenzen*

Carrasco, Figueras (1986)
Carrasco (2002)
Carrasco (2003a)
Carrasco (2003b)
Carrasco (2003c)
Carrasco (2004a)
Carrasco (2004b)
Carrasco (2005a)
Carrasco (2005b)
Carrasco (2006)
Carrasco, Sune (2007)
METFAC (2007): http://dit.upc.es/qine/tools/metfac/

2. *Projektstatus*

METFAC wurde erstmalig im Jahre 1986 als Teil einer PhD-Dissertation entwickelt (Carrasco, Figueras (1986)). Es wurde seitdem aktiv und kontinuierlich verbessert. Die neueste Version stammt von Januar 2007.

3. *Lizenztyp*

Das METFAC-Werkzeug ist kopierrechtlich geschützte Software, jedoch frei zugänglich für Einzelpersonen sowie Institute der akademischen Forschung bzw. der Grundlagenforschung zu Lehrzwecken (außer zur Beratung und Benutzung in nationalen oder internationalen

Forschungsprojekten mit voller oder teilweiser finanzierter oder personeller Unterstützung durch Gewinn-orientierte Einrichtungen).

Es werden 4 Lizenztypen angeboten:

- Eine lizenzfreie Version des Werkzeugs mit eingeschränkter Funktionalität (zeitkontinuierliche Markov-Modelle mit höchstens 500 Zuständen können generiert und gelöst werden).

- Eine lizenzfreie 30-Tage-Evaluierungsversion mit voller Funktionalität.

- Eine lizenzfreie voll-funktionsfähige Version des Werkzeugs für Einzelpersonen und Institute der Grundlagenforschung zu Lehrzwecken sowie akademischer nicht-gewinnbringender Grundlagenforschung.

- Eine reguläre Lizenz mit einer Gebühr von 5000 €, die eine voll-funktionsfähige Version des Werkzeugs für Einzelpersonen oder akademische Institute sowie Institute der Grundlagenforschung zu anderen Zwecken als denen der Ausbildung oder nicht-gewinnbringender Forschung sowie für nicht-akademische und nicht Grundlagen-orientierte Forschung betreibende Institutionen enthält.

In allen Fällen wird die Lizenz nur für eine spezielle Kombinationen des Betriebssystem und der Hardware-Plattform gewährt und darf nur auf einer einzelnen Hardware-Plattform verwendet werden, die mittels der MAC-Nummer (*Medium Access Card*) identifiziert und von der Lizenz unterstützt wird.

4. *Allgemeiner Zweck*

Analyse der Leistung, Zuverlässigkeit und Leistungsfähigkeit komplexer Systeme durch bewertete, zeitkontinuierliche Markov-Ketten-Modelle.

5. *Plattform*

Linux-Kernel 2.4.4 und darüber; Solaris 2.x; bei Bedarf jede andere Unix-Variante mit C-Shell und einem ANSI/ISO-Compiler (Standard 89/90).

6. Modell

6.1. Modellklasse

Analytisch, IT-Ebene.

6.2. Modelltyp

Finite zeitkontinuierliche Markov-Ketten-Modelle mit bewerteten zustandsabhängigen Raten.

6.3. Modellbeschreibung:

METFAC ist ein Softwarewerkzeug für die Analyse der Leistung, Zuverlässigkeit und Leistungsfähigkeit von komplexen Systemen mittels bewerteter zeitkontinuierlicher Markov-Ketten-Modelle. Es erlaubt die Spezifizierung von beliebigen finiten zeitkontinuierlichen Markov-Ketten-Modelle mit zustandsabhängigen Reward-Raten und bietet einige numerische Methoden für die Berechnung von verschiedenen Maßen an, die aus dem resultierenden stochastischen Bewertungsprozess entstehen.

6.4. Modellierte Systeme

Wiederherstellbare Systeme.

6.5. Modelleingabe

METFAC offeriert eine Modellbeschreibungssprache, die auf Ableitungsregeln basiert. Die Sprache besitzt folgende Schlüsselwörter:

- Beschreibung der Struktur:

`action, new_state, parameters, state_variables, response`

- Beschreibung von Eigenschaften:
  `production_rules, response, reward_rate, start_rate, with_prob, with_rate, initial_probability`

- Beschreibung des Programmflusses:
  `end, if, external, no, yes`

- Beschreibung von Typen:
  `int, double`

Weiterhin unterstützt die Sprache arithmetische Zuweisungen sowie relationale, binäre arithmetische, logische, inkrementelle, dekrementelle und Cast-Operatoren sowie unäres Plus und Minus.

6.6. Modellausgabe

Erwartete transiente Reward-Rate, erwartete stationäre Reward-Rate, erwartete durchschnittliche Reward-Rate, kumulative Reward-Gegenverteilung, Gegenverteilung der Intervallverfügbarkeit, erwartete kumulative Entlohnung bis zum Verlassen einer Teilmenge von Zuständen, kumulative Reward-Verteilung bis zum Verlassen einer Teilmenge von Zuständen.

6.7. Schnittstellen

Das Werkzeug verfügt in der derzeitigen Version über keine graphische Benutzeroberfläche (GUI), es wird jedoch angekündigt, dass die nächste Version eine vollständig integrierte, Windows-basierte GUI besitzen wird.

6.7.1. Eingabe

Für die Modelleingabe werden Textdateien benutzt: die Datei zur Spezifikation des

Modells (.spec) und eine optionale C-Datei (.c). Die Spec-Datei beschreibt das Modell nach der Syntax der auf Ableitungsregeln basierenden Modellbeschreibungssprache. Die C-Datei enthält Definitionen aller externen modellspezifischen Funktionen, die innerhalb der Spec-Datei verwendet werden.

6.7.2. Ausgabe

Die Ausgabe wird mittels einer zweiphasigen Modellkompilierung generiert. Im ersten Durchlauf wird die Spec-Datei in eine C-Datei übersetzt, die anschließend kompiliert und ausgeführt wird. Eine interaktive Ausführung ist möglich.

7. *Anwendungsfälle*

Zuverlässigkeitsmodell eines 5-Level-RAID-Speicherteilsystems, Zuverlässigkeitsmodell eines Speichersystems; Leistungsmodell eines Multiprozessor-Systems, GRID-Cluster-Computer-systems, Speichernetzwerkes (*Storage Area Network*) und Kommunikationsnetzwerkes.

8. *Annahmen und Einschränkungen*

Nicht bekannt.

METASAN

1. *Quellen / Referenzen*

Movaghar, Meyer (1984)
Sanders, Meyer (1986)

2. *Projektstatus*

METASAN (*Michigan Evaluation Tool for the Analysis of Stochastic Activity Networks*) ist

der Vorgänger von UltraSAN und Möbius. Die Entwicklung wurde eingestellt, die Erfahrungen sind in Möbius eingeflossen, welches derzeit stetig weiterentwickelt wird.

3. *Lizenztyp*

Unbekannt.

4. *Allgemeiner Zweck*

Evaluierung der Leistung und Zuverlässigkeit von Systemen durch Analyse und Simulation.

5. *Plattform*

UNIX

6. *Modell*

6.1. Modellklasse

Analytisch, IT-Ebene.

6.2. *Modelltyp*

Zur Modellierung werden Stochastic-Activity-Networks (SAN) eingesetzt, welche eine Erweiterung von Petri-Netzen sind.

6.3. Modellbeschreibung

Der Nutzer erstellt zur Modellierung zwei Dateien, eine für die Struktur des Modells und eine für die Daten des Experiments.

6.4. Modellierte Systeme

Wiederherstellbare und nicht-wiederherstellbare Systeme.

6.5. *Modelleingaben*

Die Spezifikation des SAN erfolgt mit Hilfe der Beschreibungssprache SANSCRIPT. Diese enthält die initiale Markierung, Abdeckung, Verteilungen der Aktivitätszeiten (Feuerzeiten), Fallverteilung und Haltebedingungen.

6.6. Modellausgaben

Leistungsfähigkeit, welches eine Metrik für Leistung und Zuverlässigkeit ist, für wiederherstellbare und nicht-wiederherstellbare Systeme.

6.7. Schnittstellen

Eingaben erfolgen über die Sprache SANSCRIPT.

7. *Anwendungsfälle*

Nicht bekannt.

8. *Annahmen und Einschränkungen*

Annahmen sind nicht bekannt, die Komplexität der Berechnungen erhöht sich schnell mit der Größe des Modells.

UltraSAN

1. *Quellen / Referenzen*

Couvillion et al. (1991)
Sanders (1995)

2. *Projektstatus*

UltraSAN wurde von der PERFORM Gruppe (*Performability Engineering Research Group*) an der University-of-Illinois-at-Urbana-Champaign bis zur Mitte der neunziger Jahre entwickelt. Die Erfahrungen mit diesem Werkzeugen sind in die Entwicklung des Nachfolgers Möbius eingeflossen. Die weitere Beschreibung entspricht dem Abschnitt über Möbius.

Möbius

1. *Quellen / Referenzen*

Clark et al. (2001)
Möbius (2007): http://www.mobius.uiuc.edu/
Sanders (2006)

2. *Projektstatus*

Das Möbius-Projekt ist eines der Hauptforschungsprojekte der PERFORM-Gruppe (*Performability Engineering Research Group*) im Center für Reliable-and-High-Performance- Computing an der University-of-Illinois-at-Urbana-Champaign. Die Forschung an Möbius wurde von Motorola, der NSF (*National Science Foundation*) und von der DARPA unterstützt. Das Möbius-Projekt basiert auf früherer Arbeiten an dem Werkzeug UltraSAN, welches zur Modellierung und Lösung von SAN (*Stochastic Activity Networks*) verwendet wurde. Das Projekt ist derzeit aktiv und unter ständiger Weiterentwicklung.

3. *Lizenztyp:*

Akademische und kommerzielle Lizenz, kostenlose 30-Tage-Lizenz für nichtakademische Organisationen.

4. *Allgemeiner Zweck*

Möbius ist ein Software-Werkzeug zur Modellierung des Verhaltens komplexer Systeme. Ursprünglich entwickelt, um Zuverlässigkeit, Verfügbarkeit und Leistungsfähigkeit von Computer- und Netzwerksystemen zu studieren, kann es neben den ursprünglichen Anwendungen für eine breite Anzahl von diskreten, ereignisorientierten Systemen (*Discrete Event Systems*) genutzt werden – von biochemischen Reaktionen innerhalb von Genen bis zu den Effekten böswilliger Angreifer auf Sicherheits-Computersysteme.

5. *Plattform*

Möbius läuft auf Windows XP oder 2000 und Linux Fedora Core 3 und später. Es werden eine JRE und ein C++-Compiler benötigt.

6. *Modell*

6.1. Modellklasse

Quantitativ – analytisch, IT-Ebene.

6.2. Modelltyp

Möbius unterstützt viele verschiedene kombinierbare, als atomar bezeichnete Modelltypen. Ursprünglich nur SAN werden inzwischen aber auch traditionelle Modellierungs-paradigmen, wie Warteschlangentheorie oder stochastische Petri-Netze, und auch neuere Formalismen, wie stochastische Prozessalgebren (SPA), stochastische Automaten-netzwerke, Fehlerbäume und kombinatorische Blockdiagramme, unterstützt.

6.3. Modellbeschreibung

Alle diese Modelle können miteinander kombiniert und hierarchisch verknüpft werden, da die Entwickler davon ausgehen, dass nicht ein Modellierungsformalismus für jedes Problem die optimale Beschreibung bieten kann. Möbius bietet eine offene Plattform, in

der Formalismen einfach integriert, verknüpft und zusammen ausgewertet werden können. Zudem gibt es zwei unterschiedliche Techniken zur Lösung der Modelle: Diskrete Ereignissimulation und zustandsbasierte analytische / numerische Techniken.

6.4. Modellierte Systeme

Beispielsweise fehlertolerante Systeme; durch seine erweiterbare Struktur ist Möbius in der Lage, alle Systeme auszuwerten und zu analysieren, für die Modelle integriert wurden.

6.5. Modelleingabe

Modellabhängig.

6.6. Modellausgabe

Modellabhängig, vor allem Zuverlässigkeit, Verfügbarkeit, Sicherheit und Leistung des Systems.

6.7. Schnittstelle

Möbius besitzt eine graphische Benutzeroberfläche (Modelleditor), mit der die Modelle erstellt und verknüpft werden können. Externe atomare Formalismen können hinzugefügt werden.

7. Anwendungsfälle

Das Werkzeug unterstützt die Validierung von Systemen in verschiedenen Anwendungsbereichen, wie:

- Systeme der Informationstechnologie und Netzwerke
- Draht- und drahtlose Telekommunikationssoftware- und Hardware-Systeme
- Systeme der Luft- und Raumfahrt
- kommerzielle und staatliche Sicherheits-Informationssysteme und -Netzwerke

- biologische Systeme

8. *Annahmen und Einschränkungen*

   Es können nur Markov-Modelle verwendet werden. Deshalb sind nur exponentiell verteilte oder augenblicklich erfolgende Transitionen erlaubt. Der Startzustand des Modells muss stabil sein.

NFTAPE

1. *Quellen / Referenzen*

   Stott (2000)
   Stott et al. (2000)
   Stott et al. (2002)

2. *Projektstatus*

   Das Projekt wurde an der University-of-Illinois-at-Urbana-Champaign entwickelt. Die letzte Veröffentlichung stammt aus dem Jahre 2002. NFTAPE ist ein Nachfolger von FTAPE, dessen Entwicklung bis in die 90er Jahre zurückreicht.

3. *Lizenztyp*

   Vermutlich existiert eine akademische Lizenz.

4. *Allgemeiner Zweck*

   NFTAPE (*Networked Fault Tolerance and Performance Evaluator*) ist eine Software-basierte Umgebung zur automatischen Bestimmung der Zuverlässigkeit eines Netzwerkes auf Basis der Fehlerinjektionsmethode. NFTAPE ermöglicht es dem Benutzer, einen Plan zur Fehlerinjektion aufzustellen, die Experimente auf Basis dieses Plans durchzuführen und

abschließend die gewonnen Ergebnisse zu analysieren. Mittels einer Skriptsprache können verschieden Verfahren zur Fehlerinjektion beschrieben und auch zur Laufzeit umgeschaltet werden. NFTAPE verfügt über verschiedene Maße zur Verlässlichkeit wie beispielsweise Zuverlässigkeit, Verfügbarkeit und Fehleraufdeckung (*Coverage*). NFTAPE kann auf eine verteilte Umgebung und auf verschiedene Zielsysteme angewendet werden und arbeitet aufwendungsarm auf dem Zielsystem.

5. *Plattform*

Solaris, Linux.

6. Modell

6.1. *Modellklasse*

Quantitativ – Benchmarking und Test, IT-Ebene.

6.2. Modelltyp

Software-basierte Fehlerinjektion.

6.3. Modellbeschreibung

NFTAPE besitzt eine Komponenten-basierte Architektur und gliedert sich in die Module Lightweight-Fault-Injector (LWI), Lightweight-Fault-Trigger und einer Kontrollumgebung. Mittels Debugger oder Treibern können vom Anwender Fehler in den Speicher (Hauptspeicher, Register, etc.), in das Betriebssystem, in I/O-Geräte, in Netzwerkkarten und in Controllern der Zielplattform injiziert werden. Auf der Zielplattform läuft ein Prozessmanager, der die Befehle für die Fehlerszenarien von der Kontrollumgebung empfängt und umsetzt.

6.4. Modelliertes System

Verteilte Systeme (Netzwerke).

6.5. Modelleingabe

Modellierte Fehlerszenarien.

6.6. Modellausgabe

Numerisch: Verlässlichkeit (Zuverlässigkeit, Verfügbarkeit, Fehleraufdeckung und Wiederherstellung).

6.7. *Schnittstellen*

6.7.1. Eingabe

Python-Skripte (Jython).

6.7.2. Ausgabe

Graphisch.

7. *Anwendungsfälle*

NFTAPE wurde in folgenden Bereichen eingesetzt:

- Motorola IDEN MicroLite: Test eines Base-Station-Controller im digitalen Mobiltelefonnetzwerk.
- DHCP (*Dynamic Host Configuration Protocol*): Auswertung der Überprüfung des Kontrollflusses einer Anwendung
- SIFT-Umgebung (*Software Implemented Fault Tolerance*) für die REE-Testeinbettung (*Remote Exploration and Experimentation*): REE ist ein Projekt, um

COTS-basierte Parallelrechner zur Datenanalyse auf einem Raumfahrzeug einzusetzen.

- Server-Anwendungen für das Internet: Evaluierung der Schadensanfälligkeit von SSH- und FTP-Diensten durch Fehlerinjektion.

8. *Annahmen und Einschränkungen*

Das Werkzeug verfügt nicht über eine abstrakte Beschreibungssprache für Fehlerinjektions-Szenarien. Keine Modifikation des Quellcodes und Skalierung möglich.

NUMAS

1. *Quellen / Referenzen*

Beilner et al. (1989)
Mueller (1984)

2. *Projektstatus*

Die letzten bekannten Entwicklungen stammen aus dem Jahre 1989.

3. *Lizenztyp*

Unbekannt.

4. *Allgemeiner Zweck*

NUMAS ist ein Leistungsanalyse-Tool, bei dem jeder Knotenpunkt mit Fehlertoleranz-Charakteristiken erweitert werden kann.

5. *Plattform*

Sun.

6. Modelle

6.1. Modellklasse

Analytisch, IT-Ebene.

6.2. Modelltypen

Warteschlangennetze, Markov-Ketten.

6.3. Modellbeschreibung

Ein modelliertes System wird als Warteschlangennetz dargestellt und dann in eine Markov-Kette umgewandelt. Diese wird mit der Berechnung der stationären Verteilung unter Verwendung der schrittweisen Summierung, des Gaußschen Eliminations- und des Gauß-Seidel-Verfahrens numerisch gelöst.

6.4. Modellierte Systeme

Wiederherstellbare und nicht-wiederherstellbare Systeme.

6.5. Modelleingabe

- Multi-Server-Warteschlangen und lastabhängige Ausfallquoten für Server,
- Kundenverteilung über die Warteschlangen-Knotenpunkte,
- Abstufungsmodus (*Degradation Mode*) jedes Knotenpunkts.

6.6. Modellausgabe

Stationäre Leistung.

6.7. Schnittstellen

6.7.1. Eingabe

NUMAS kann über zwei Schnittstellen verwendet werden:

- Interaktive NUMAS-Schnittstelle
- HI-SLANG Modellierungssprache (HIT Systemsprache)

Beide Methoden erlauben die Eingabe von Modellen von herabstufbaren High-Level-Warteschlangennetzen. Zudem stellt HI-SLANG die hierarchische Modellierung zur
Verfügung.

6.7.2. Ausgabe

Die stationäre Leistung wird in tabellarischer Form ausgegeben. Diese ist hinsichtlich der Eingabeelemente sortiert.

7. Anwendungsfälle

Nicht bekannt.

8. Annahmen und Einschränkungen

Es ist nicht möglich Abhängigkeiten im Ausfall- oder Reparaturprozess verschiedener Knotenpunkte zu modellieren. Nur stationäre Messungen können berechnet werden.

OpenSESAME

1. Quellen / Referenzen

OpenSesame (2007): http://www.lrr.in.tum.de/~walterm/OpenSesame/
Walter et al. (2007)

2. *Projektstatus*

Das Projekt wird aktiv weiterentwickelt – die letzte Veröffentlichung stammt aus dem Jahre 2007.

3. *Lizenztyp*

Akademische Lizenz; kommerzielle Lizenz auf Nachfrage.

4. *Allgemeiner Zweck*

OpenSESAME (*Simple but Extensive Structured Availability Modeling Environment*) kombiniert die benutzerfreundliche Modellierung von kombinatorischen Techniken (RBD) mit der Ausdrucksstärke zustandsbasierter Modellierungstechniken (Petri-Netze). Als weiteres Feature kann auch die gegenseitige Abhängigkeit von Komponenten spezifiziert werden.

5. *Plattform*

Linux/Unix mit folgenden Tools:

- Der Petri-Netz-Löser DSPNexpress-NG, der bei OpenSESAME mitgeliefert wird.
- GNU gcc/g++ version 3.2
- GNU make version 3.79.1
- GNU scientific library (gsl) version 0.9 (gsl-config must be in your binary path)
- BYACC Version 1.8 - Berkeley YACC, nicht Bison benutzen.
- FLEX Version 2.5.4
- RPCGEN
- CSH / SSH (auch dann, wenn derselbe Rechner für Modellierung und Auswertung verwendet wird)

- JAVA Runtime Environment (JRE) Version 1.4

6. Modell

6.1. *Modellklasse*

Quantitativ – analytisch, IT-Ebene.

6.2. *Modelltyp*

Zuverlässigkeitsblockdiagramme (*Reliability Block Diagrams*), Fehlerabhängigkeits-diagramme (*Failure Dependencies Diagrams*), stochastische Petri-Netze.

6.3. *Modellbeschreibung*

Die vom Benutzer eingegebenen RBD werden intern in eine semantisch äquivalente Beschreibung mittels stochastischer Petri-Netze umgewandelt und mittels des Tools DSPN-express-NG gelöst.

6.4. *Modelliertes System*

Redundantes System.

6.5. *Modelleingabe*

Als Eingabe spezifiziert der Benutzer Zuverlässigkeitsblockdiagramme des zu analysierenden Systems, die im weiteren Verlauf schrittweise spezifiziert werden können. Die Komponenten eines RBD können mit Verteilungen sowie Ausfall- und Wiederherstellungsraten (im einfachsten Fall MTTF und MTTR) versehen werden. Komponentenabhängigkeiten können ebenfalls hinzugefügt oder entfernt werden. Weitere Eingabe: Verschiedene Redundanzschemen und Wiederherstellungsstrategien, Ausfall-abhängigkeiten.

6.6. *Modellausgabe*

Ausfälle mit allgemeiner Ursache, Ausfallvorhersage (durch Zerstörung oder Blockierung von Komponenten), Non-perfect-Ausfallerkennung, Non-Zero-Failover-Times bei redundanten Systemen im Stand-by-Betrieb, begrenzte Reparaturkapazitäten.

6.7. *Schnittstellen*

6.7.1. *Eingabe*

Graphisch, tabellarisch (Variablentabelle).

6.7.2. *Ausgabe*

Graphisch.

7. *Anwendungsfälle*

OpenSESAME kann zur Modellierung von fehlertoleranten Hochverfügbarkeitssystemen verwendet werden, wie beispielsweise Webserver, Telekommunikationsschaltstellen, zuverlässige Energielieferanten oder komplette Netzwerke.

8. *Annahmen und Einschränkungen*

Die klassischen Auswertemethoden für Fehlerbäume und RBD können nicht angewendet werden.

PENELOPE

1. Quellen / Referenzen

Mauser (1990)

Meer, Sevcikova (1996)
Meer, Sevcikova (1997)
Zadach (1998)

2. Projektstatus

Das Werkzeug wurde in Meer, Sevcikova (1996) beschrieben. Im selben Jahr erschien die noch immer aktuelle Version 3.1. Der derzeitige Projektstatus ist allerdings unbekannt.

3. Lizenztyp

Unbekannt (vermutlich existiert eine akademische Lizenz).

4. Allgemeiner Zweck

PENELOPE ist ein Werkzeug zur Leistungsanalyse und -optimierung. Es basiert auf der Theorie der erweiterten Extended-Markov-Reward-Modelle (EMRM), wie sie von Meer beschrieben wurden. In Mauser (1990) wird es spezifiziert und implementiert. Zwei Jahre später wurde das Werkzeug um Algorithmen zur stationären und Optimierung erweitert. Des Weiteren wurden verschiedene Berechnungsstrategien zur algorithmischen Analyse der Leistungsfähigkeit hinzugefügt.

5. Plattform

SUN-4 mit Sun OS 4/5 i.V. mit X Window System Release 11 Version 4 oder höher sowie OSF/Motif GUI System Version 1.1 oder höher.

6. Modell

6.1. *Modellklasse*

Quantitativ – analytisch, IT-Ebene.

6.2. *Modelltyp*

Extended Markov Reward Model (EMRM), Controlled Stochastic Petri Nets (COSTPN), Optimierung und Simulation.

6.3. *Modellbeschreibung*

EMRMC sind eine von Mauser weiterentwickelte Version der Markov-Reward-Modelle, die auf in den 60er Jahren untersuchten Markovschen Entscheidungsprozessen beruhen. Sie enthalten zusätzlich Rekonfigurationskanten und Verzweigungszustände. COSTPN sind eine Erweiterung der Stochastic-Reward-Netze um zusätzliche Transitionen (beispielsweise zur Rekonfiguration). Zur numerischen Auswertung werden die COSTPN in EMRMC übersetzt.

6.4. *Modelliertes System*

Dynamisch rekonfigurierbare Systeme.

6.5. *Modelleingabe*

Modellparameter unbekannt.

6.6. *Modellausgabe*

Transiente Optimierung, stationäre Optimierung von Strategien; Simulation; transiente Analyse, stationäre Analyse (Leistungsmaße).

6.7. *Schnittstellen*

6.7.1. *Eingabe*

C-Implementierung (nicht verfügbar), graphische Benutzeroberfläche.

6.7.2. Ausgabe

C-Implementierung (nicht verfügbar), in Dateien, graphisch.

7. *Anwendungsfälle*

Beispielstudie eines Bereitstellungsmodells für den Gefahrenfall (*Emergency Supply Model*):
Es wird von einem Lager mit endlicher Kapazität ausgegangen. Ist das Lager leer und kommen Kunden, deren Nachfrage nicht befriedigt werden kann, so werden deren Anfragen in einer Warteschlange festgehalten. Die Nachbestellungen werden exponentiell und unabhängig voneinander modelliert. Das System wird mittels COSTPN modelliert und EMRMC werden daraus abgeleitet. Mittels PENELOPE lässt sich die optimale Strategie für das Modell ermitteln.

8. *Annahmen und Einschränkungen*

Analyse und Optimierung auf beschränkte Zeithorizonte (Missionszeiten).

PENPET

1. *Quellen / Referenzen*

Haverkort, Niemegeers (1996)
Lepold (1991)

2. *Projektstatus*

Das Werkzeug PENPET wurde von Lepold et. al. an der Siemens AG in Kooperation mit der Universität Mulhouse Anfang der 90er Jahre entwickelt. Der derzeitige Projektstatus ist unbekannt.

3. *Lizenztyp*

   Unbekannt.

4. *Allgemeiner Zweck*

   Das Werkzeug PENPET wurde zur Analyse der Leistungsfähigkeit und Zuverlässigkeit von fehlertoleranten Systemen entwickelt.

5. *Plattform*

   C-Implementierung unter UNIX, in Verbindung mit X-Window-System und MOTIF.

6. *Modell*

   6.1. Modellklasse

   Quantitativ – analytisch, qualitativ, IT-Ebene.

   6.2. Modelltyp

   GSPN (*Generalized Stochastic Petri Nets*), Markovsche Reward-Modelle.

   6.3. Modellbeschreibung

   PENPET erlaubt die Beschreibung von High-Level-Modellen als Cluster von Low-Level-Modellen. Der Zuverlässigkeitsteil des Modells wird auf dem höchsten Level mit Hilfe einer strukturellen Formel beschrieben. Es werden Modelle (auch Makromoleküle genannt) definiert, die aus einer bestimmten Anzahl von Teilen von Submodellen (Molekül-Cluster) oder Komponenten (Atome) bestehen. Die Beschreibung in Textform mittels der Strukturformel ähnelt einer chemischen Beschreibung von Molekülen. Für jedes Teil eines Submodells oder einer Komponente wird festgelegt, welche Anzahl

vorhanden ist sowie die Anzahl derjenigen Komponenten, die für einen Betrieb der nächsthöheren Ebene des hierarchischen Modells notwendig sind. Ausfallraten und Fehlerabdeckung werden ebenfalls für jedes Submodell oder jede Komponente gespeichert. Komponenten und Submodelle werden mit Hilfe von GSPN beschrieben, die in Bibliotheken verfügbar sind. Von der Strukturformel des Systems wird ein GSPN abgeleitet, welches die generellen Aspekte der Zuverlässigkeit beschreibt. Von diesem GSPN werden alle möglichen Systemzustände sowie die darunter liegenden Markov-Ketten abgeleitet. Der Leistungsteil wird mit Hilfe einer Reward-Strategie (Markovsche Reward-Modelle) beschrieben. Umgekehrt lassen sich auch Zuverlässigkeitseigenschaften (beispielsweise Fehlerraten im strukturellen Zustandsmodell) aus der errechneten Entlohnung ableiten.

6.4. Modelliertes System

Keine Angaben.

6.5. Modelleingabe

Strukturformel des Systems; benötigte GSPN-Teilmodelle, falls nicht in der Bibliothek vorhanden.

6.6. Modellausgabe

Maße zur Leistungsfähigkeit (momentan und stationär).

6.7. Schnittstellen

Die Eingabe erfolgt in Textform, die Ausgabe tabellarisch oder graphisch.

7. Anwendungsfälle

PENPET wurde zur Analyse der Leistungsfähigkeit von fehlertoleranten Multiprozessor-Systemen verwendet. Genauere Angaben sind nicht verfügbar.

8. *Annahmen und Einschränkungen*

   Nicht bekannt.

Relex Reliability Studio: PRISM

1. *Quellen / Referenzen*

   Relex (2007): http://www.relex.com

2. *Projektstatus*

   Das Projekt wird aktiv weiterentwickelt.

3. *Lizenztyp*

   Kommerzielle Lizenz, Evaluierungslizenz (Demoversion).

4. *Allgemeiner Zweck*

   Reliability Studio ist eine Werkzeugzusammenstellung der Firma Relex, um Aussagen über die Zuverlässigkeit, Leistung und Wartbarkeit von Computersystemen zu treffen. Es werden Methoden und Vorgehensmodelle zur Evaluierung der Kritikalität von Fehlern und sowie Modelle zur Analyse der Zuverlässigkeit und Verfügbarkeit angeboten. Mittels Optimierung und Simulation können umfangreiche statistische Aussagen über diese Kriterien angeboten werden. PRISM ist ein Standard für die MTBF-Vorhersage und Zuverlässigkeitsanalyse, der von RAC (*Reliability Analysis Center*) stammt. PRISM wurde in die Werkzeug-Suite Reliability Studio integriert (zur Verbesserung von RBD, Fehlerbäumen, FMEA und anderen Modulen).

5. *Plattform*

PC/Windows.

6. Modell

6.1. Modellklasse

Quantitativ – analytisch, qualitativ, IT- und Prozess-Ebene.

6.2. Modelltyp

Markov-Modelle, Zuverlässigkeitsblockdiagramme (RBD) mit Monte-Carlo-Simulation, FMEA/FMECA, FTA/ETA, FRACAS und Human-Factor-Risk-Analysis.

6.3. Modellbeschreibung

Mittels Zustandsübergangsdiagrammen für Markov-Ketten werden betriebsbereite, eingeschränkt betriebsbereite und fehlerhafte Zustände und deren Übergänge dargestellt. Es werden graphische Methoden bereitgestellt. Daneben verwendet Reliability Studio unter anderem folgende Methoden: Risikoanalyse (FMEA, FMECA, HF-PFMEA, Fehlerbäume) und Prozesskontrolle (FRACAS, Zuverlässigkeitsvorhersage) – siehe dazu die Be-schreibung der Methoden.

6.4. Modelliertes System

Nicht-wiederherstellbare und wiederherstellbare Systeme, redundante Systeme.

6.5. Modelleingabe

Markov-Modelle: Funktionale Zusammenhänge mittels Zustandsübergangsdiagrammen.

Risikoanalyse: Prozessbeschreibung.

FMEA / FMECA: Workflow.

6.6. Modellausgabe

Je nach Modell oder Methode: Verfügbarkeit, Zuverlässigkeit, (mittlere) Nichtverfügbarkeit, MTBF, MTTF, Ausfallraten, erwartete Anzahl an Fehlern, Gesamtausfallzeit, Ausfallhäufigkeit, Gefahrenrate (Hazard Rate).

Markov-Modelle: Transiente und stationäre Verfügbarkeitsmaße, Kapazität, Ausfallhäufigkeit, Kosten sowie Anzahl der Besuche eines bestimmten Zustands.

RBD: Stationäre und zeitabhängige Ergebnisse, Konfidenzintervalle.

FMEA/FMECA: Kritikalitätsmatrix und -ordnung, Risikoniveau, Risikoprioritätszahl.

Risikoanalyse: Risk-Assessment-Calculations.

6.7. Schnittstellen

Web-basiert, graphisch und tabellarisch (Excel, Acces, Text), Report-Formate (Excel, PDF, HTML).

7. Anwendungsfälle

Anwendungsgebiete sind unter anderem Luft- und Raumfahrt, Automobil, Öl und Gas, Medizintechnik, Eisenbahn und Telekommunikation. Umfangreiche Anwendungsfallstudien sind über die Webseite nachlesbar.

8. Annahmen und Einschränkungen

Möglicherweise ist die Wahl der Plattform nur auf PC/Windows beschränkt. Ansonsten keine bekannt.

QUAKE

1. *Quellen / Referenzen*

   Tixeuil et al. (2006)

2. *Projektstatus*

   Das Werkzeug wurde an der Universität Coimbra in Portugal entwickelt, es existiert eine Zusammenarbeit mit INRIA France.

3. *Lizenztyp*

   Unbekannt.

4. *Allgemeiner Zweck*

   QUAKE ist ein Werkzeug zur Bestimmung der Zuverlässigkeit von Grids und Web-Services.

5. *Plattform*

   Keine Angaben.

6. *Modell*

   6.1. Modellklasse

   Quantitativ – Benchmarking und Test, IT-Ebene.

   6.2. Modelltyp

Fehlerinjektion, Zeitreihenanalyse (*Time-series Analysis*), Benchmark.

6.3. Modellbeschreibung

QUAKE besteht aus dem BMS (*Benchmark Management System*) und einem SOAP-Webserver – dem SUT (*System Under Test*). Das BMS besteht aus Modulen zur automatischen Durchführung des Benchmarks (Definition, Ausführung und Speicherung der Resultate).

Der Webserver wird unter einer bestimmten Arbeits- und Fehlerlast (*Workload / Faultload*) betrieben. Von verschiedenen Clients werden Anfragen an den Server über SOAP-XML simuliert. Alle Maschinen sind über das NTP (*Network Time Protocol*) synchronisiert. Neben SOAP sind auch andere Middleware-Ansätze verwendbar.

Für das Benchmarking werden verschiedene Modi verwendet: Im Learning-Modus werden grundlegende Daten zur Leistung gesammelt. Im Arbeitslast-Modus wird der Server unter erhöhter Arbeitsbelastung getestet (Stresstest). Zusätzlich kann das Verhalten des Servers durch Fehlerinjektion in den Systemressourcen des Servers untersucht werden (Arbeitslast / Fehlerlast-Modus).

Die Arbeitslast wird mittels folgender Verteilungen generiert: Kontinuierliche maximale Arbeitsbelastung, stationäre Verteilung, maximale zeitlich begrenzte Arbeitsbelastung, Surge-Load-Szenario (konstante Requestanzahl mit einzelnen Peaks).

Die Fehlerlast wird durch ein externes Programm induziert. Dabei werden folgende Systemressourcen belastet: Hauptspeicherverbrauch nach Ramp-Up-Verteilung, Erzeugung einer hohen Anzahl an Threads, extensive Benutzung von File-Deskriptoren, Verbrauch von Datenbankverbindungen.

6.4. Modelliertes System

Verteilte Systeme (Client-Server-Architekturen mit Internet-Verbindung).

6.5. Modelleingabe

Mittels Prozeduren und Regeln beschriebene Arbeits- und Fehlerlasten.

6.6. Modellausgabe

- Ausfälle, Charakterisierung (Stillstand (*Hung-up*), Absturz (*Crash*), Zombie-Server), von Clients beobachtete Ausfälle (Logdateien).

- Grundlegende Leistungsmaße:
  Durchsatz in Anfragen pro Sekunde, durchschnittliche Antwortzeit bei einer Anfrage, Conforming Requests, Funktionalität des Service, Turn-Around-Zeiten.

- Leistungsmaße bei einer bestimmten Stress-Arbeitsbelastungsverteilung sowie zusätzlich mit Fehlerinjektion auf der Serverseite.

- Zuverlässigkeitsbasierte Maße:

  - Integrität (*Integrity*): Anzahl an Fehlern in den Daten des Webservices (Datenbank) bei Arbeitsbelastung (*Workload*) und Fehlerbelastung (*Faultload*).
  - Verfügbarkeit: Zeit, in der das System für Anfragen während des Testlaufs zur Verfügung steht.
  - Autonomie: Manuelles Eingreifen ist nötig, da der Server sich in einem Hang-Up-Zustand befindet – im Gegensatz zu einem Crash, bei dem der Server selbständig hochfahren kann (*Reboot*).
  - Effektivität der Selbstreparatur (*Self Healing Effectiveness*).

6.7. Schnittstellen

In Textform (Eingabe) oder numerisch (Ausgabe). Details sind nicht verfügbar.

7. Anwendungsfälle

Anwendungen aus dem Entwicklungsbereich (Grids und Webservices). Details sind nicht verfügbar.

8. *Annahmen und Einschränkungen*

   Keine bekannt.

Reliability Center: PROACT, LEAP

1. *Quelle / Referenz*

   Reliability Center (2007): http://www.reliability.com

2. *Projektstatus*

   Der Status des Projektes ist aktuell.

3. *Lizenz*

   Kommerzielle Lizenz.

4. *Allgemeiner Zweck*

   PROACT (*PReserving Event Data, Ordering the Analysis Team, Analyzing Event Data, Communicating Findings & Recommendations Tracking For the Bottom-Line Results*) ist der Name einer gleichnamigen Softwarelösung für das Proact-Prozessmodell, einer Variante der Root-Cause-Analysis-Methode (RCA-Methode). PROACT bietet die Möglichkeit, diesen Prozess regelkonform in einem Unternehmen zu etablieren.

   LEAP ist ein Werkzeug zur Umsetzung der FMEA- und Opportunitäts-Methode. Der Anspruch von LEAP ist es, die 20% aller Ereignisse zu finden, die die Ursache für 80% des

Verlustes sind.

5. *Plattform*

Vermutlich PC/Windows (kommerziell).

6. *Modell*

6.1. Modellklasse

Quantitativ – analytisch, qualitativ, IT-Ebene.

6.2. Modelltyp

PROACT: Prozessmodell nach der Root-Cause-Analysis-Methode.
LEAP: FMEA und Opportunitätsanalyse.

6.3. Modellbeschreibung

Die RCA-Methode gehört zu den analytisch-qualitativen Methoden. Mittels eines Baumdiagramms (*Logic Tree*) werden Ursachen und deren Auswirkungen untersucht. Es werden die Bereiche Prozess, technische Ausrüstung und menschliche Verfügbarkeit untersucht. Die FMEA-Methode ist eine analytisch-qualitative Methode und die Opportunitätsanalyse ist die Analyse aufgezeichneter Daten.

6.4. Modelliertes System

Technische Systeme, Unternehmensstrukturen.

6.5. Modelleingabe:

PROACT: Beschreibung der Ursachen eines Prozesses (physikalische und menschliche) und deren Auswirkungen.

LEAP: Wahrscheinlichkeitsdaten und historische Daten.

6.6. Modellausgabe

PROACT: Daten über die Effektivität des Prozesses, Empfehlungen.

LEAP: Ranking der unerwünschten Ereignisse.

6.7. Schnittstellen

In Textform (Eingabe) und graphisch, online, Tabellen-basiert in der Ausgabe.

7. *Anwendungsfälle*

Anwendungsbereiche sind unter anderem: Luft- und Raumfahrt (NASA), Eisenbahn (Amtrak), Lebensmittelindustrie (Bacardi), Multikonzerne (General Electric), Automobil (General Motors), Elektrizitätswerke (Virginia Powers), Öl und Gas (Shell).

8. *Annahmen und Einschränkungen*

Keine bekannt, vermutlich nur für PC/Windows.

Reliass

1. *Quellen / Referenzen*

Reliass (2007): http://www.reliability-safety-software.com

2. *Projektstatus*

Die Software und die Webseite sind relativ aktuell. Textreferenzen sind bezüglich der

Betriebssysteme veraltet (Windows 98/ME/NT/2000).

3. *Lizenztyp*

Kommerzielle Lizenz, CD-ROM mit Demoversionen.

4. *Allgemeiner Zweck*

Reliass hat sich auf den Verkauf von e-Commerce-Software für Web-basierte Anwendungen spezialisiert, zudem bieten sie Skripte für Computersicherheit an. Folgende Werkzeugketten werden von Reliass vertrieben:

- +ASENT Toolkit: Softwarebasierte Zuverlässigkeits- und Wartbarkeitsanalyse
- EAGLE Toolkit: Standardisiertes Softwaresystem zur Logistikunterstützung
- AIMSS: Verwaltung von technischen Informationen komplexer Systeme
- +Logan Fault Tree: Fehlerbaum- und Ereignisablaufanalyse
- Logan Monte Carlo: Informationen nicht verfügbar
- +Raptor Simulation: Zuverlässigkeitsanalyse mittels RBD
- +RAMP: Simulation von Prozess-basierten Systemen
- +RAM Commander: Toolkit zur Zuverlässigkeits- und Wartbarkeitsanalyse
- D-LCC: Berechnung von Lebenszykluskosten
- +MEADEP: Daten-basierte Zuverlässigkeitsanalyse
- FavoWeb: Werkzeug nach der FRACAS-Methode
- FMEA-Pro6: Werkzeug zur FMEA
- PHA-Pro6: Werkzeug zur Gefahrenanalyse (Hazard Analysis)
- SVA-Pro: Werkzeug zur Schadenpotentialanalyse
- PRISM: Werkzeugzusammenstellung zur Zuverlässigkeitsvorhersage

Im Folgenden werden nur die Werkzeuge mit einem '+' beschrieben, da sie im Kontext dieser Zusammenstellung relevant erscheinen.

5. *Plattform*

IBM PC mit Windows.

6. Modell

6.1. Modellklasse

Quantitativ – analytisch, IT-Ebene.

6.2. Modelltyp

Es werden folgende Modelle (Methoden) unterstützt:

- FMECA, RCM und Testbarkeitsanalyse (ASENT, RAM Commander),
- fehlerbehebende und vorbeugende Wartung, Thermal-Analysis, Sensitivitätsanalyse (ASENT),
- RBD (ASENT, Raptor Simulation) mit Monte-Carlo-Simulation (RAM Commander), RBD und Markov-Ketten (MEADEP)
- Fehlerbaum- und Ereignisablaufanalyse (Logan Fault Tree, RAM Commander)
- Weibull-Ausfallverteilungen (RAMP)
- Zuverlässigkeitsvorhersage (RAM Commander)
- Weak-link-Analysis und Phased-Simulation (Raptor Simulation)

FMECA, RCM, RBD, FTA sind im Dokument 'Methodenbeschreibung' beschrieben.

6.3. Modellbeschreibung

RAM Commander: Die RBD in Verbindung mit der Monte-Carlo-Simulation ist eine Erweiterung der RBD durch Simulation, falls keine analytische Lösung berechenbar ist, wie beispielsweise bei komplexen Systemen mit Teilbereitschaft, Stand-by, aktiver Redundanz, eingeschränkter Reparatur oder voneinander abhängigen RBD-Elementen.

MEADEP: Das Werkzeug besteht aus 4 Modulen: Datenvorverarbeitung, Dateneditor und Analyse; Modellgenerator und Modellauswerter. Die Ergebnisse werden entweder

direkt aus den Daten oder von Zuverlässigkeitsmodellen abgeleitet. Bei der Ableitung aus vorhandenen Daten werden diese mittels des Datenvorverabeitungsmoduls importiert und anschließend mittels Zuverlässigkeitsblockdiagrammen und Markov-Ketten modelliert. Sind keine Daten vorhanden, können mit Hilfe des Modellgenerators auch Zuverlässigkeitsmodelle erstellt werden.

6.4. Modelliertes System

Wiederherstellbare, redundante und komplexe Systeme (gemischt parallel/seriell).

6.5. Modelleingabe

Systemmodell, Zuverlässigkeitsmodell (seriell/parallel), Systemdaten.

6.6. Modellausgabe

Numerische Werte (MTBCF, MTTR, Man-Hours) mit Zuverlässigkeitsmodell (parallel, seriell).

6.7. Schnittstellen

6.7.1. Eingabe:

Die Eingabe erfolgt zumeist graphisch über Editoren.
MEADEP: ASCII, Import von Datenbanken unter anderem ACCESS, dBASE und Paradox.

6.7.2. Ausgabe:

LSAR-Werkzeuge. Exportmöglichkeit der Ergebnisse besteht in der Regel zu Microsoft Software wie Excel, Word oder Datenbanken.

7. Anwendungsfälle

FavoWeb wurde von Lockheed Martin für das JSF-Projekt (F-35 Kampfflugzeug) verwendet. Dazu existiert auf der FavoWeb-Homepage ein externer Link. Darüber hinaus sind keine weiteren Anwendungsfälle bekannt, da sich die Seite mit den aufgeführten Geschäftskunden im Aufbau befindet.

8. *Annahmen und Einschränkungen*

   Keine Einschränkungen bekannt. Möglicherweise ist das Werkzeug nur für Plattformen mit PC/Windows geeignet.

Reliasoft

1. *Quellen / Referenzen*

   Reliasoft (2007): http://www.reliasoft.com

2. *Projektstatus*

   Die Produkte und die Webseite sind aktuell.

3. *Lizenztyp*

   Kommerzielle Lizenz sowie eine kostenlose Evaluierungslizenz. Es existieren verschiedene Typen der kommerziellen Lizenz:

   - Einzelbenutzerlizenz (*Single User Licence*)
   - Netzwerklizenz (*Standard and Concurrent Network Licence*)
   - unbegrenzte Benutzerlizenz
   - mietbare Lizenz

4. *Allgemeiner Zweck*

Die Werkzeuge dienen dazu, den Bereich der Verfügbarkeitsanalyse von Software (bis auf MPC 3 und Lambda Predict) abzudecken. Die Produktpalette umfasst folgende Werkzeuge:

- Weibull++: Lebensdaten- oder Weibull-Analyse
- ALTA (*Accelerated Life Testing Data Analysis*): Datenanalyse aus Stresstest
- RGA (*Reliability Growth Analysis*): Analyse der Verbesserung der Qualität eines Softwareproduktes bei der prototypischen SW-Entwicklung
- BlockSim: Zuverlässigkeits-, Verfügbarkeits- und Wartbarkeitsanalyse eines Systems mittels Zuverlässigkeitsblockdiagrammen oder der Fehlerbaummethode
- Xfmea: FMEA und FMECA
- RCM++: RCM
- MPC 3: Wartbarkeitsanalyse von technischen Systemen aus der Luftfahrt nach der MSG-3-Norm
- Lambda Predict: Zuverlässigkeitsvorhersage (*Reliability Prediction Analysis*),
- Reno: Visuelle, stochastische Ereignissimulation von physikalischen, finanziellen oder organisatorischen Modellen
- XFACAS: Analyse der Zuverlässigkeit nach der FRACAS-Methode

5. Plattform

PC mit Microsoft Windows NT/2000/XP.

6. Modell

6.1. Modellklasse

Quantitativ – analytisch, qualitativ, IT-Ebene.

6.2. Modelltyp

Lebensdatenanalyse, RCM, FMEA/FMECA, RCM, RBD, FRACAS.
Stochastische Ereignissimulation.

6.3. Modellbeschreibung

Die folgenden Punkte beziehen sich lediglich auf das Werkzeug (RENO) zur stochastischen Ereignissimulation.

Die stochastische Ereignissimulation erfolgt mittels Ablaufsdiagramm-Modellen. Es gibt vorgefertigte Blöcke (*Constructs*) und Variablen (*Definitions*). Die Blöcke werden vom Anwender zur Modellierung verschiedener Szenarien genutzt, welche durch Simulation gelöst werden können. Beispielsweise können Blöcke zur Verzweigung mit If-Then-Anweisungen aus Verhaltensbeschreibungen, die in Form von Text vorliegen, eingefügt werden. Des Weiteren existieren Blöcke für logische Gatter und zur Speicherung. Eine Zerlegung in Unterblöcke ist ebenfalls möglich. Variablen können vom Benutzer definiert werden und sind während des Projektes global verfügbar. Sie enthalten neben Konstanten auch Tabellen und Funktionen sowie Wahrscheinlichkeiten und Verteilungen von Zufalls-variablen.

6.4. Modelliertes System

Physikalische, wirtschaftliche und organisatorische Systeme.

6.5. Modelleingabe

System -oder Ablaufbeschreibung eines Prozesses.

6.6. Modellausgabe

Statistische Angaben über Zuverlässigkeit und Verfügbarkeit.

6.7. Schnittstellen

Die Ablaufsdiagramm-Modelle werden manuell über einen Editor eingegeben. Die Ausgabe der Simulationsergebnisse erfolgt dann tabellarisch im Editorfenster oder im

Excel-Tabellenformat.

7. *Anwendungsfälle*

Stochastische Ereignissimulation umfasst folgende Anwendungsgebiete:

- System- und Produkterstellung (Zuverlässigkeits- und Ausfallanalyse, Risiko- und Gefahrenanalyse, Schadenpotentialanalyse, konzeptionelles und probabilistisches Design, Systementwicklung, Fertigung, Materialbeschaffung, Qualitätskontrolle)

- Umwelteinflüsse sowie Umweltressourcenplanung und -steuerung (Ökosystem-modellierung)

- Wirtschaftsmodellierung, wie beispielsweise strategische Planung, Geschäftsprozess-modellierung, Finanzanalyse, Risikoanalyse und -steuerung, Kostenmodellierung, Portfoliomanagement.

Zusätzlich werden auf der Webseite einige Anwendungsfälle zur Zuverlässigkeits- und Risikoanalyse sowie aus dem Bereich Optimierung vorgestellt. Es gibt beispielsweise folgende Anwendungsfälle, unter anderem für die Zuverlässigkeitsanalysen: Zuverlässigkeit von Schneemaschinen und Waschmaschinen.

8. *Annahmen und Einschränkungen*

Keine Einschränkungen bekannt. Möglicherweise ist das Werkzeug nur für Plattformen mit PC/Windows geeignet.

SAVE

1. *Quellen / Referenzen*

Conway, Goyal (1986)

Goyal et al. (1986a)

Goyal et al. (1986b)

Goyal et al. (1987)

2. *Projektstatus*

SAVE ist basierend auf den Werkzeugen ARIES, CARE III und HARP entstanden. Das Projekt ist momentan nicht aktiv, jedoch werden die angebotenen Monte-Carlo-Techniken in mehreren kommerziellen Verfügbarkeitswerkzeugen benutzt.

3. *Lizenztyp*

Unbekannt. Die letzte Implementierung stammt aus dem Jahre 1987.

4. *Allgemeiner Zweck*

Lösen von probabilistischen Modellen der Systemverfügbarkeit und -zuverlässigkeit von einsatzorientierten und kontinuierlich betriebsbereiten Systemen.

5. *Plattform*

FORTRAN 77.

6. *Modell*

6.1. *Modellklasse*:

Analytisch, Simulation, IT-Ebene.

6.2. Modelltyp

Homogene Markov-Ketten.

6.3. Modellbeschreibung:

Die stationäre Verfügbarkeit wird durch Lösen der homogenen Mengen von zeitgleichen linearen Gleichungen berechnet, die aus der Markov-Kette abgeleitet werden. Die Empfindlichkeit wird in Abhängigkeit von Parametern der Übergangsraten (Fehler- und Reparaturrate) berechnet, indem die linearen Gleichungen differenziert werden, welche der Markov-Kette genügen. Die mittlere Zeit bis zum Ausfall ist aus dem transienten Verhalten des Systems ableitbar. Die Modelle können auch direkt durch Simulation gelöst werden, in dem direkte und analoge Monte-Carlo-Methoden verwendet werden.

6.4. Modelliertes System

Wiederherstellbare und nicht-wiederherstellbare Systeme.

6.5. Modelleingabe

- Lösungsmethoden: Numerisch, Markov, kombinatorisch

- Komponenten geordnet nach Typ (beispielsweise Prozessoren, Datenbanken und Ersatz); für jede Komponente ist ein Ersatz über die Ersatzausfallrate definiert; Betriebszuverlässigkeiten für Komponenten.

- Ausfallraten für Komponenten (im ruhenden und betriebsbereiten Zustand), Reparaturraten für Komponenten, Reparaturabhängigkeiten, Ausfallmodi für das System, Wahrscheinlichkeit des Ausfallmodus für jede Komponente; Spezifizierung der betroffenen Komponenten für jeden Ausfallmodus; Spezifizierung der Bedingungen, unter denen das System betriebsbereit ist. Wiederherstellungsstrategien (Reihenfolge, in der die Komponenten wiederhergestellt werden).

6.6. Modellausgabe

Stationäre Verfügbarkeit, Empfindlichkeitsanalyse, MTTF.

6.7. Schnittstellen

6.7.1. Eingabe

Zur Spezifizierung der Eingabemodelle wird eine Stapelverarbeitungssprache verwendet.

6.7.2. Ausgabe

Nicht verfügbar, vermutlich werden die Ergebnisse in eine Datei geschrieben. Deshalb genügt eine Schnittstelle zum Dateisystem (Handler), um mit dem Werkzeug zu kommunizieren.

7. *Anwendungsfälle*

Computer in der Luft- und Raumfahrt (nicht-wiederherstellbare Systeme) und Telefonschaltsysteme, Mehrzweckcomputer, Transaktionssysteme (wiederherstellbare Systeme)

8. *Annahmen und Einschränkungen:*

Verwendet lediglich die Exponentialverteilung, berücksichtigt nicht transiente und intermittierende Fehler.

SHARPE 2000/2002

1. *Quellen / Referenzen*

Hirel et al. (2000)
Puliafito et al. (1997)
Sahner, Trivedi (1986a)
Sahner, Trivedi (1986b)

Sahner at el. (2000)
Sharpe (2007): http://www.ee.duke.edu/~kst/software_packages.html
Trivedi (2002)
Trivedi (2007)

2. *Projektstatus*

Die erste Version des SHARPE-Werkzeugs war in Sahner, Trivedi (1986a) beschrieben. Die Folgeversion aus Sahner, Trivedi (1986b) basierte auf den Werkzeugen SPADE, DEEP und HARP. Das Werkzeug wird stetig weiterentwickelt und existiert aktuell unter der Bezeichnung SHARPE 2000 (Hirel et al. (2000)) und unter SHARPE 2002 (Trivedi (2002)) sowie mit Weberweiterungen (Puliafito et al. (1997)).

3. *Lizenztyp*

SHARPE 2000/2002 ist frei erhältlich zur akademischen Nutzung (http://shannon.ee.duke.edu/tools/agreement_sharpe.htm).

Eine kommerzielle Lizenz für SHARPE 2000/2002 existiert, jedoch sind deren Bedingungen nicht einsehbar.

4. *Allgemeiner Zweck*

SHARPE 2000/2002 ist eine Werkzeugzusammenstellung. Darin sind eine Spezifikationssprache und Lösungsmethoden der meisten aller gewöhnlich benutzten Modelltypen zur Modellierung von Leistung, Zuverlässigkeit und Leistungsfähigkeit enthalten.

5. *Plattform*

Windows95, Windows98, WindowsNT, Windows2000.
Linux, Solaris.
Java, kann auf jeder verfügbaren JVM ausgeführt werden.

6. Modell

6.1. Modellklasse

Analytisch, Simulation, IT-Ebene.

6.2. Modelltyp

Markov-Ketten (irreduzible, azyklische und phasentypische), Semi-Markov-Ketten, Zuver-lässigkeitsblockdiagramme, Fehlerbäume, Zuverlässigkeitsgraphen, Single-Chain-Product-Form-Warteschlangennetze, Multi-Chain-Product-Form-Warteschlangennetze; genera-lisierte, stochastische Petri-Netze, seriell-parallele Graphen.

6.3. Beschreibung

Das Werkzeug ermöglicht die Spezifizierung und Analyse von Modellen zur Leistung, Zuverlässigkeit und Leistungsfähigkeit. Die Modelltypen ermöglichen eine kombinatorische Auswahl einerseits von Fehlerbäumen, Warteschlangennetzwerken und Zustandsräumen und andererseits von Markov- und Semi-Markov-Bewertungsmodellen sowie stochastischen Petri-Netzen. Stationäre und transiente Maße sowie Intervall-Maße können berechnet werden. Die Ausgabewerte (Maße) eines Modells können als Parameter für ein anderes Modell verwendet werden. Diese Eigenschaft ermöglicht eine hierarchische Kombination von verschiedenen Modelltypen. Tabelle 1.1 gibt einen Überblick der Modelle und der Einsatzmöglichkeiten.

	Zuverlässigkeit	***Leistungs-fähigkeit***	***Performabilität***
Fehlerbaum	X		
Multizustands-Fehlerbaum	X		
Zuverlässigkeitsblock-diagramm	X		
Zuverlässigkeitsgraph	X		
Markov-Kette	X	X	X
Semi-Markov-Kette	X	X	X
Markov-regenerativer Prozess	X	X	X
Generalisierte stochastische Petri-Netze (GSPN)	X	X	X
Stochastische Netze mit Entlohnung (Stochastic Reward Net)	X	X	X
Product-form-Queueing-Networks		X	
Multiple-Chain-Product-Form- Queueing-Networks		X	
Aufgabengraph (Task Graph)		X	
Schrittweise arbeitende Systeme	X		

Tabelle 1.1: Modellmatrix SHARPE 2000

6.4. Modelliertes System

Nicht-wiederherstellbare und wiederherstellbare Systeme.

6.5. Modelleingabe

- Markov-Ketten mit absorbierenden Zuständen (azyklisch oder PH-typisch): Übergänge und Übergangsraten; Bewertungen; Wahrscheinlichkeiten der Initialzustände.

- Irreduzible Markov-Ketten: Übergänge und Übergangsraten; Bewertungen; Wahrscheinlichkeiten der Initialzustände (nur für transiente Analyse).

- Azyklische Semi-Markov-Ketten: Übergänge und Übergangsverteilungen; Bewertungen; Wahrscheinlichkeiten der Initialzustände.

- Irreduzible Semi-Markov-Ketten: Übergänge und Übergangsverteilungen; Bewertungen.

- Syntax der Eingabe im Zuverlässigkeitsblockdiagramm:
  block *name*{(*param_list*)} <blockline> end.
  Eine *blockline* kann weiterhin spezifiziert werden zu: *component* (Name und Exponentialverteilung); serielle und parallele Kombination von Komponenten, K-aus-N-Systeme haben identische Komponenten, K-aus-N-Systeme haben verschiedene Komponenten.

- Syntax der Eingabe bei Fehlerbäumen:
  ftree name {(*param_list*)} <ftreeline> end.
  Eine *ftreeline* kann weiterhin spezifiziert werden zu: Verschiedene Typen von Ereignissen mit zugewiesenen exponentiell polynomial; sich wiederholenden Ereignissen, versetzten Ereignissen, AND-Gatter, OR-Gatter, K-aus-N-Gatter mit identischen Eingaben, K-aus-N-Gatter mit verschiedenen Eingaben.

- Zuverlässigkeitsgraphen:
  Gerichtete Kanten (Namen und exponentielle polynomiale kumulative Verteilungsfunktion für die Zeit-bis-zum-Ausfall-Eigenschaft des Pfades), ungerichtete Kanten (Namen und exponentielle polynomiale kumulative Verteilungsfunktion für die Zeit-bis-zum-Ausfall-Eigenschaft beider Pfade)

- Single-Chain-Product-Form-Queuing-Networks:
  Station-zu-Station-Wahrscheinlichkeiten, Stationstypen und Parameter, Anzahl von Kunden pro Kette.

- Multiple-Chain Product-Form-Queueing-Networks:
  Station-zu-Station-Wahrscheinlichkeiten, Stationstypen und Parameter, Anzahl von Kunden pro Kette.

- Generalisierte stochastische Petri-Netze (GSPN):
  Stellen und initiale Anzahl von Marken; Namen der zeitabhängigen Transitionen, Typen und Raten, Namen sofort schaltender Transitionen und Gewichte, Stellen-zu-Transitionen-Pfeile und Vielfachheiten, Hemmende Pfeile und Vielfachheiten.

- Serielle-parallele Graphen:
  Kanten, exponentiell polynomial für jeden Knoten des Graphen, Ausgangstypen für jeden gegebenen Knoten (parallele Teilgraphen sind probabilistisch, alle parallelen Teilgraphen müssen vollständig sein, einer der Teilgraphen muss vollständig sein); Kantenwahrscheinlichkeiten, Multipfad-Informationen.

6.6. Modellausgabe

Im Allgemeinen sind die folgenden Ausgaben an verschiedene Modellebenen gebunden: Zuverlässigkeit der gewählten Komponenten, Systemzuverlässigkeit über einen Intervall, stationäre Systemverfügbarkeit. Die detaillierte Ausgabe erfolgt weiter unten.

- Kumulative Verteilungsfunktion:
  Seriell-parallele azyklische gerichtete Graphen: Graph-Ausführungszeit; falls Multipfad-Informationen gefordert sind, wird die kumulative Verteilungsfunktion für jeden Pfad angegeben.

- Zuverlässigkeitsblockdiagramm, Fehlerbaum, Zuverlässigkeitsgraph:
  Falls es sich bei der Funktion, angewendet auf jede Komponente, um die kumulative Verteilungsfunktion handelt, so ist die Ausgabe die kumulative Verteilungsfunktion der Systemausfallzeit. Sollte aber die Funktion, angewendet auf jede Komponente, die momentane oder stationäre Verfügbarkeit sein, so ist die Ausgabe die momentane oder stationäre Verfügbarkeit des Systems.

- Azyklische Semi-Markov-Ketten, azyklische und phasentypische Markov-Ketten:
  Falls der Name eines Zustandes nicht angegeben wird, wird die kumulative Verteilungsfunktion für die Zeit ausgegeben, bis ein absorbierender Zustand erreicht wurde. Falls ein Zustand angegeben wird, wird die kumulative Verteilungsfunktion für die Zeit ausgegeben, bis dieser Zustand erreicht wurde, sowie die Wahrscheinlichkeit diesen Zustand zu erreichen. Falls ein transienter Zustand erreicht wurde, so wird die transiente Wahrscheinlichkeitsfunktion für das Erreichen dieses Zustands ausgegeben.

- Irreduzible Markov-Kette:
  Transiente Wahrscheinlichkeitsfunktion für den Aufenthalt in einem gegebenen Zustand.

- Nicht-irreduzible generalisierte stochastische Petri-Netze:
  Zeit bis zur Erreichung einer absorbierenden Markierung.

- Bedingte Wahrscheinlichkeit dafür, dass ein Zustand innerhalb vorgegebener Zeit verlassen wurde, Mittelwert und Varianz der kumulativen Verteilungsfunktion; Wahrscheinlichkeit, einen Zustand zu erreichen.

- Wahrscheinlichkeit, dass die gesammelte Bewertung zur Zeit der Absorbierung weniger oder gleich dem vorgegebenen Wert ist (Markov- und Semi-Markov-Modelle mit bewerteten Zuständen).

- Generalisierte stochastische Petri-Netze (GSPN):
  Die durchschnittliche Anzahl von Markierungen an spezifizierten Platzierungen; Wahrscheinlichkeit, dass die Stelle leer ist, Auslastung und Durchsatz einer Transition (stationär, zu gegebener Zeit – symbolisch und numerisch, Zeitdurchschnitt).

- Product-Form-Queuing-Networks: Durchsatz, durchschnittliche Antwortzeit, Auslastung (Single- und Multi-Chain-Networks).

6.7. Schnittstellen

6.7.1. Eingabe

SHARPE 2000/2002 akzeptiert Eingaben mit Hilfe der SHARPE Stapelverarbeitungssprache oder über eine Java-basierte graphische Benutzeroberfläche (GUI). Die Stapelverarbeitungssprache besitzt eine vordefinierte Syntax und Semantik, die eine Erzeugung, Modifizierung und Analyse hierarchischer Modelle erlauben. Die GUI unterstützt dieselben Optionen. Programmierbare, externe Eingabeverbindungen zu SHARPE sind deshalb am ehesten über die Stapelverarbeitungssprache herzustellen, obwohl vermutlich auch Elemente Swing-basierter Benutzerschnittstellen von externen Programmen angesteuert werden können. Eine Web-basierte Schnittstelle für SHARPE ist ebenfalls verfügbar.

6.7.2. Ausgabe

SHARPE 2000/2002 stellt Simulationsergebnisse über die Java-basierte GUI dar. Nichtgraphische Ergebnisse sind mittels der Stapelverarbeitungssprache verfügbar, wobei die Ergebnisse auch in Form von Textdateien empfangen werden können. Der Export von Modellen und Analyseergebnissen wird innerhalb der GUI unterstützt (beispielsweise in Excel-, JPEG- oder EPS-Dateien). Eine Web-basierte Schnittstelle für SHARPE ist ebenfalls verfügbar.

7. Anwendungsfälle

Auf der Projektwebseite wird behauptet, dass Softwarepakete an 280 Orten installiert sind, allerdings gibt es keine Belege für diese Aussage.

8. Annahmen und Einschränkungen

Nicht verfügbar.

SPNP

1. *Quellen / Referenzen*

   Ciardo et al. (1989)
   SPNP (2007a): http://www.ee.duke.edu/~kst/software_packages.html
   SPNP (2007b): http://www.ee.duke.edu/~chirel/MANUAL/manual.pdf

2. *Projektstatus*

   Das Softwarepaket wurde von Ciardo und anderen an der Duke-University entwickelt Ciardo et al. (1989). Seit 2001 sind keine neuen Versionen mehr veröffentlicht worden.

3. *Lizenztyp*

   Akademische sowie kommerzielle Lizenz auf Nachfrage.

4. *Allgemeiner Zweck*

   Das SPNP (*Stochastic Petri Net Package*) ist ein vielfach einsetzbares Modellierungswerkzeug zur Analyse der Leistung, Zuverlässigkeit und Leistungsfähigkeit komplexer Systeme.

5. *Plattform*

   Version 6 wurde für MS-DOS sowie Solaris und Linux kompiliert.

6. *Modell:*

   6.1. Modellklasse:

   Quantitativ – analytisch, IT-Ebene.

6.2. *Modelltyp:*

SRN (*Stochastic Reward Net*), FSPN (*Fluid Stochastic Petri Nets*).

6.3. Modellbeschreibung

Der Modelltyp, der für die Eingabe verwendet wird, ist ein SRN, das verschiedene strukturelle Erweiterungen für GSPN beinhaltet, wie Markenabhängigkeit (z.B. markenabhängige Transitionen, Kardinalitäten und Wächter), und das erlaubt, Reward-Raten mit jeder Markierung zu verbinden. Die Reward-Funktion kann ebenfalls markenabhängig sein. Sie werden mittels der Sprache CSPL (C-basierte SRN-Sprache) spezifiziert, die eine Erweiterung der Programmiersprache C durch Hinzufügen von Konstrukten zur Beschreibung von SRN-Modellen ist. SRN-Spezifikationen werden automatisch in Markov-Reward-Modelle konvertiert, die anschließend analytisch-numerisch gelöst werden, um die Vielfalt transienter, stationärer und kumulativer Maße sowie von Empfindlichkeitsmaßen zu berechnen. Für SRN mit absorbierender Markierung können die mittlere Absorptionszeit und die erwartete akkumulierte Entlohnung bis zur Absorption berechnet werden. In der neuesten Version lassen sich auch FSPN und Nicht-Markov-SPN spezifizieren. Diese werden mittels der diskreten Ereignissimulation gelöst.

6.4. Modellierte Systeme

Unbekannt.

6.5. Modelleingabe

SRN

6.6. Modellausgabe

Numerische Ergebnisse der Simulation, Erreichbarkeitsgraph, Markov-Kette oder

Markov-Prozess (je nach SRN), Beschreibung des Petri-Netzes.

6.7. Schnittstellen

6.7.1. Eingabe

Mittels der Sprache CSPL wird das SRN spezifiziert.

6.7.2. Ausgabe

Textdateien.

7. *Anwendungsfälle*

Es existieren verschiedene exemplarische Anwendungsbeispiele:

- Molloy-Beispiel
- Software-Leistungsanalyse
- M/M/m/b-Warteschlangen
- Verfügbarkeitsanalyse für Datenbanksysteme
- Kanalwiederherstellung in Mobilfunknetzen
- ATM-Netzwerke bei Überladung

8. *Annahmen und Einschränkungen*

Keine bekannt.

SoftRel LLC: FRESTIMATE

1. *Quellen / Referenzen*

Neufelder (2000a)

Neufelder (2000b)
SoftRel (2007): http://www.softrel.com/

2. *Projektstatus*

Das Projekt wird stetig weiterentwickelt, jedoch ist der Veröffentlichungstermin der neuesten Version weiterhin unbekannt.

3. *Lizenztyp*

Das Werkzeug gibt es in drei Versionen, deren Preise vom Funktionsumfang abhängig sind: Frestimate Standard Edition ($1999), Frestimate Manager's Edition ($1149) sowie Frestimate Metrics Package ($249).

4. *Allgemeiner Zweck*

FRESTIMATE wurde zur Einschätzung der Softwarezuverlässigkeit hinsichtlich der zu erwartenden Anzahl von Fehlern pro 1000 Codezeilen entwickelt. Diese beruht auf der Überwachung des Projektmanagements und lässt sich schrittweise durch Isolierung der kritischsten Produktionsschritte (z.B. Mangel an geeigneten Tests) innerhalb des Softwareentwicklungsprozesses ausführen. Weiterhin können die Verbesserungskosten jedes möglichen Schrittes und die daraus eventuell resultierenden Vorteile gewichtet werden. Dadurch werden die Prozessverbesserungen bestimmt, die zur größten Zuverlässigkeits-verbesserung mit dem geringsten Aufwand führen.

5. *Plattform*

Windows 2000, Windows XP.

6. *Modell*

6.1. Modellklasse

Analytisch, Software.

6.2. Modelltyp

Angepasstes Modell, basierend auf Korrelation.

6.3. Modellbeschreibung

Es wird ein angepasstes Modell verwendet. Die Eingaben des Werkzeugs erhält man durch Einschätzung betriebswirtschaftlicher und organisatorischer Verfahren innerhalb des Softwareentwicklungsprozesses mittels einer Untersuchung. Es wird der Korrelationsfaktor zwischen den Untersuchungsergebnissen und der vorhandenen Wissensdatenbank (Stand vom November 2006: 85 Anwendungsfälle) berechnet. Eine Abschätzung der Projektzuver-lässigkeitsleistung, beruhend auf dem Korrelationsfaktor, ist als passendster Anwendungsfall der Wissensdatenbank gegeben. Es werden Zuverlässigkeitsleistungs-metriken ausgegeben (Fehler auf 1000 Codezeilen, MTTF, Verfügbarkeit, Zuverlässigkeit, Ausfallhäufigkeiten).

6.4. Modelliertes System

Wiederherstellbare Systeme, Softwaresysteme.

6.5. Modelleingabe

Die Antworten der Untersuchung bilden die Modelleingabe. Abhängig von der Version des Werkzeugs kann die Untersuchung 15 bis 80 Fragen umfassen.

6.6. Modellausgabe

Die Ausgabe erfolgt über verschiedene Diagramme und Tabellen, die die erwarteten Fehler auf 1000 Codezeilen, MTTF, Verfügbarkeit, Zuverlässigkeit und Ausfallhäufigkeiten anzeigen.

Diagramme zur Kostenabschätzung durch Veränderungen von Teilen des Softwareproduktionsprozesses und durch relative Auswirkungen von Verfügbarkeitsverbesserung.

6.7. Schnittstellen

6.7.1. Eingabe

Die Eingabe erfolgt über eine graphische Benutzeroberfläche in Form einer Untersuchung.

6.7.2. Ausgabe

Die Benutzeroberfläche ermöglicht das Laden und Speichern des Modells.

7. Anwendungsfälle

Laut Angaben der Firma nutzen mehr als 600 Kunden das Werkzeug. Sie stellen die Untersuchungsstatistiken aus ihrer Wissensdatenbank zur Verfügung. In den meisten Fällen wurde das Werkzeug für Softwareentwicklungen in der Halbleiter-, Verteidigungs-, Luft- und Raumfahrtindustrie verwendet, allerdings sind keine Firmennamen aufgelistet.

8. Annahmen und Einschränkungen

Es wird angenommen, dass die untersuchte Firma oder das System sich genauso verhalten wird wie Firmen, deren Profile in der Datenbank enthalten sind. Da die Anzahl der Firmen in der Datenbank recht klein ist (nur 85; für einige Industrien existiert genau ein repräsentatives Muster), könnte die Genauigkeit dieses Werkzeuges und der Methode niedrig sein. Das ist besonders dann der Fall, wenn der Studienbereich nicht durch die Datenbank abgedeckt wird (beispielsweise hat das FRESTIMATE keinen einzigen Eintrag in der Wissensdatenbank für die Entwicklung von Banksoftware).

SURE

1. *Quelle/Referenz*

   Butler (1986a)
   Butler (1986b)
   Butler, Johnson (1995)
   Butler (2007)

2. *Projektstatus*

   Für das SURE Projekt (Semi-Markov Unreliability Range Evaluator) sind seit 1995 keine Veränderungen mehr verzeichnet.

3. *Lizenztyp*

   Unbekannt.

4. *Allgemeiner Zweck*

   Zuverlässigkeitsanalyse von fehlertoleranten Architekturen. Berechnet untere und obere Grenzen für die Wahrscheinlichkeit, dass das System läuft oder ausgefallen ist.

5. *Plattform*

   Sun SPARC, Linux, Beta für Windows 98.

6. Modell

   6.1. Modellklasse:

   Analytisch, IT-Ebene.

6.2. Modelltyp

SURE benutzt ein Semi-Markov-Modell zur Modellierung.

6.3. Modellbeschreibung

SURE ist ein Zuverlässigkeits-Analyseprogramm, das zur Berechnung von oberen und unteren Schranken für die Zustandswahrscheinlichkeiten, dass das System betriebsbereit oder ausgefallen ist, innerhalb einer großen Klasse von Semi-Markov-Modellen entwickelt wurde. Das SURE-Bounding-Theorem besitzt algebraische Lösungen, die sich infolge-dessen sehr effizient auch für größere und komplexere Systeme berechnen lassen.

6.4. Modellierte Systeme:

Wiederherstellbare und nicht-wiederherstellbare Systeme.

6.5. Modelleingaben

Anstelle der Fehlerbaum oder PMS-Beschreibung als Eingabe wurde eine abstrakte Sprache entwickelt Butler (1986a), um die Menge an Regeln zur Erzeugung der Zustandsübergangsmatrix zu definieren. Diese Sprache wurde im Programm ASSIST implementiert, welches in PASCAL geschrieben wurde und auf einer VAX 111750 läuft. ASSIST ist eigentlich unabhängig von SURE und kann auch mit anderen Werkzeugen verwendet werden. Die Sprache besteht aus fünf Typen von Anweisungen:

a) *Constant-definition*-Anweisung
b) SPACE-Anweisung
c) START-Anweisung
d) DEATHIF-Anweisung
e) TRANTO-Anweisung

Eine *Constant-definition*-Anweisung gleicht einem Bezeichner mit alphanumerischen Zeichen oder Zahlen. Die SPACE-Anweisung spezifiziert den Zustandsraum, auf dem das Markov-Modell als n-dimensionaler Vektor definiert ist, wobei jede Komponente des Vektors eine Eigenschaft des zu modellierenden Systems definiert. Die START-Anweisung zeigt an, welcher Zustand der Startzustand des Modells ist und welche die internen Zustände des Systems repräsentieren. Die DEATHIF-Anweisung spezifiziert die toten Zustände (*Death* oder *Trapping States*) innerhalb des Modells. Die TRANTO-Anweisung wird benutzt, um alle Übergänge im Modell rekursiv zu generieren.

6.6. Modellausgaben

Obere und untere Schranken der Wahrscheinlichkeit des totalen Systemausfalls, Wahrscheinlichkeitsschranken für jeden toten Zustand im Modell, Liste jedes Pfades durch das Modell und die Wahrscheinlichkeit des Durchlaufens.

6.7. Schnittstellen:

SURE hat eine graphische Benutzeroberfläche auf Windows 98 für Eingaben in der verwendeten Sprache und zur graphischen Aufbereitung der Ergebnisse.

7. *Anwendungsfälle*

Wiederherstellbare und nicht-wiederherstellbare Systeme.

8. *Annahmen und Einschränkungen*

Annahmen sind nicht bekannt. Der Fokus liegt auf Zuverlässigkeit und Verfügbarkeit. Wartungsstrategien können nicht in die Modelle mit einbezogen werden.

SURF-2

1. *Quelle/Referenz*

Kanoun, Blain (1996)
Kanoun, Borrel (2007)

2. Projektstatus

Die Version 3.0 wurde 1997 veröffentlicht. Seitdem sind keine Updates mehr erschienen. SURF-2 ist der Nachfolger von SURF.

3. Lizenztyp:

Es gibt eine kommerzielle Lizenz von der LAAS-CNRS. Für akademische Zwecke ist auch eine preisgünstige Lizenz erhältlich.

4. Allgemeiner Zweck

SURF-2 ist ein allgemeines Werkzeug zur Bestimmung der Verlässlichkeit von Hardware und Software basierend auf der Konstruktion, Validierung und numerischen Lösung von Markov-Ketten. Die Modellierung erfolgt entweder mit Markov-Ketten oder GSPN (*Generalized Stochastic Petri Nets*).

5. Plattform

SUN-4, SPARCstation-4, SPARCstation-5, UltraSPARC.
SUN OS 4.1.x oder SOLARIS 2.x.
X-Window (X11R5).

6. Modell

6.1. Modellklasse

Analytisch, IT-Ebene.

6.2. *Modelltyp*

Als Modell wird GSPN genutzt, eine Erweiterung von Petri-Netzen. Damit kann ein System graphisch mit seinen Zuständen beschrieben werden. Zur Evaluierung des Systems wird das Netz in eine entsprechende Markov-Kette umgewandelt.

6.3. *Modellbeschreibung*

Bei SURF-2 wird zunächst das System entweder mit Markov-Ketten oder stochastischen Petri-Netzen modelliert. Durch Berechnung von Invarianten kann das Modell validiert werden. Die mathematische Auswertung des Modells erfolgt automatisch und liefert Kennzahlen für die Zuverlässigkeit des Systems.

6.4. Modellierte Systeme

Wiederherstellbare und nicht-wiederherstellbare Systeme.

6.5. *Modelleingabe*

SURF-2 benötigt das GSPN oder eine Markov-Kette als Eingabe.

6.6. Modellausgabe

Maße für die Zuverlässigkeit einer Architektur. Reward-Strukturen können in das Verhaltensmodell eingebunden werden, um kombinierte Maße für Zuverlässigkeit, Leistung und Kosten zu erhalten.

6.7. *Schnittstellen*

SURF-2 stellt eine graphische Benutzerschnittstelle zur Verfügung, mit der das GSPN oder die Markov-Kette erstellt und verändert werden können.

7. Anwendungsfälle

Nicht bekannt.

8. *Annahmen und Einschränkungen*

Annahmen sind nicht bekannt. Beschränkungen sind die allgemeinen Grenzen beim Lösen von Markov-Ketten, das heißt, eine Zustandsexplosion beim Lösen von größeren Modellen.

Sydvest

Sydvest bietet die Werkzeuge CARA-FaultTree und Sabaton an.

2.2.1.7 CARA-FaultTree

1. *Quellen / Referenzen*

Cara-Faulttree (2007): http://www.sydvest.com/Products/Cara/prod_info.htm

2. *Projektstatus*

CARA-FaultTree wird von der Firma Sydvest entwickelt und vertrieben. Das Programm scheint nicht mehr weiterentwickelt worden zu sein. Die letzten Neuigkeiten auf der Webseite datieren von 2001.

3. *Lizenztyp*

Kommerzielle Lizenz. Eine Testversion mit begrenzter Funktionalität existiert ebenfalls.

4. *Allgemeiner Zweck*

Das Werkzeug ist nur auf die Behandlung von Fehlerbäumen spezialisiert. Die Anwendbarkeit ist dabei auf die Anwendbarkeit der Fehlerbaumanalyse begrenzt.

5. *Plattform*

Windows 9x, Windows NT, Windows 2000.

6. *Modell*

6.1. Modellklasse

Analytisch, IT-Ebene.

6.2. *Modelltyp*

Fehlerbaumanalyse.

6.3. Modelliertes System

Nicht-wiederherstellbare Systeme.

6.4. Modelleingabe

Standard für Fehlerbäume.

6.5. Modellausgabe

Minimale Schnittmengen, durchschnittliche Verfügbarkeit, Überlebenswahrscheinlichkeit, MTTF, Häufigkeit von TOP-Ereignissen, Ausfallhäufigkeitsverteilung, sechs Maße der Komponentenbedeutsamkeit (Birnbaums Zuverlässigkeit, Birnbaumsche Struktur, Vesely Fussell, Bedenklichkeitsbedeutsamkeit, Verbesserungspotential, Ordnung der kleinsten Schnittmenge).

6.6. Schnittstellen

6.6.1. Eingabe

Textdateien, die gemäß der Herstellerrichtlinien oder über eine graphische Benutzeroberfläche formatiert wurden, dienen als Eingabe. Zur Verbesserung der Lesbarkeit von großen Bäumen wird der Fehlerbaum in Seiten gegliedert.

6.6.2. Ausgabe

Die Ausgabe erfolgt über eine graphische Benutzerobfläche. Zudem ist es möglich, die Berichte als RTF-Dateien zu exportieren oder in die Windows-Ablage zu kopieren.

7. *Anwendungsfälle*

CARA-FaultTree wird von Jardine Technology verwendet. Diese ist eine Beratungsfirma, die ihren Kunden beim Lösen von Vermögensverwaltungs- und Risikomanagementproblemen in der Chemie-, Energie-, Verkehrs-, Benzin- und Ölindustrie unterstützt. Es ist jedoch noch unklar, in welchem Umfang und in welchen Projekten dieses Werkzeug genutzt wird.

8. *Annahmen und Einschränkungen*

Es ist auf Fehlerbaummodelle beschränkt.

2.2.1.8 SABATON

1. *Quellen / Referenzen*

Sabaton (2007): http://www.sydvest.com/Products/Sabaton/

2. *Projektstatus*

Sabaton wird von der Firma Sydvest entwickelt und vertrieben. Die letzten Neuigkeiten auf der Homepage datieren von 2005.

3. *Lizenztyp*

   Kommerzielle Lizenz (2390e). Eine Testversion mit begrenzter Funktionalität existiert ebenfalls.

4. *Allgemeiner Zweck*

   SABATON unterstützt FMEA (*Failure Mode and Effects Analysis*) und FMECA (*Failure Mode, Effects and Criticality Analysis*), die üblicherweise in der Produkt- und Systementwicklung zur Aufdeckung von möglichen Ausfällen und Ausfallarten sowie zur Abschätzung der Ausfallfolgen verwendet werden. Die Analyseergebnisse werden typischerweise in Designverbesserungsvorschlägen formuliert und sind auf die Beseitigung von Systemausfällen oder die Folgenminderung der Komponentenausfälle gerichtet.

5. *Plattform*

   Windows 9x, Windows NT, Windows 2000.

6. *Modell*

   6.1. Modellklasse

   Analytisch, IT-Ebene.

   6.2. Modelltyp

   FMEA/CA.

   6.3. Modellierte Systeme

   Nicht bekannt.

6.4. Modelleingabe

Standard für FMEA oder FMECA.

6.5. Modellausgabe

Die Risiko-Prioritätszahl (RPZ) wird berechnet und ausgegeben ebenso wie andere festgelegte Maße. Sie eignet sich zur Priorisierung von potentiellen Ausfällen und wird zur Klassifizierung von potentiellen Designmängeln und / oder Haftungsfragen verwendet. Die Kritikalitätsanalyse wird mittels der Kritikalitäts- und Risikomatrix durchgeführt.

6.6. Schnittstellen

6.6.1. Eingabe

Textdateien, die gemäß der Herstellerrichtlinien oder durch eine graphische Benutzeroberfläche formatiert wurden.

6.6.2. Ausgabe

Die Ausgabe erfolgt mittels einer graphischen Benutzeroberfläche, über das flexible Auswertungsressourcen aktiviert werden. Des Weiteren können darüber definierbare Vorlagen von Analyseberichten festgelegt werden. Darüber hinaus lassen sich die Berichte im PDF-Format zur elektronischen Verteilung abspeichern.

7. Anwendungsfälle

Die Anwendung des Werkzeugs besteht in zwei Fällen, allerdings ist es nicht bestimmbar, in welchem Umfang oder für welche Zwecke es genutzt wird
ISDEFE ist eine spanische Firma, die Technik- und Beratungsservices für fortgeschrittene Technologien im Verteidigungssektor und zivilen Umfeld anbietet. Das INSHT (*National Institute of Safety and Hygiene at Work*) ist die fachkundige Wissenschafts- /

Technikabteilung der Staatsregierung. Ihr Auftrag ist die Analyse und die Studie der Arbeitsschutzbedingungen.

8. *Annahmen und Einschränkungen*

SABATON eignet sich nur für FMEA- und FMECA-Modelle.

TANGRAM

1. *Quellen / Referenzen*

Berson et al. (1987)
Page et al. (1989)
Haverkort, Niemegeers (1996)

2. *Projektstatus*

TANGRAM ist eine hauptsächlich objektorientierte Werkzeugumgebung, die von Berson und anderen an der University-of-California in Los Angeles Ende der 80er Jahre entwickelt wurde. Der aktuelle Projektstatus ist unbekannt.

3. *Lizenztyp*

Unbekannt (vermutlich existiert eine akademische Lizenz).

4. *Allgemeiner Zweck*

TANGRAM liefert eine schichtartige Werkzeugumgebung, die recht einfach auf spezielle Anwendungsbereiche zugeschnitten werden kann, vorausgesetzt die geeigneten Evaluierungs-techniken sind verfügbar.

5. *Plattform*

SUN-3 unter UNIX, C- und objektorientierte PROLOG-Implementierung.

6. Modell

6.1. *Modellklasse*

Abhängig vom Einsatzgebiet.

6.2. *Modelltyp*

Abhängig vom Einsatzgebiet.

6.3. *Modellbeschreibung*

TANGRAM ist eine sehr allgemeine Modellierungsumgebung, die hauptsächlich für die Analyse der Leistungsfähigkeit, Zuverlässigkeit sowie Markovscher Entlohnung verwendet werden kann. Modelle werden als Ansammlung von Objekten spezifiziert. Objekte sind parametrisierte Instanzen von Objekttypen. Jeder Objekttyp hat einen internen Zustand, der sich mit dem Abschluss interner Ereignisse verändert. Interne Ereignisse können auch das Versenden von Nachrichten an andere Objekte auslösen. Der Gesamtstatus des Modells ist eine Zusammenstellung der internen Zustände aller Objekte. Mit jedem Zustand kann auch ein Wert (Reward) verknüpft werden. Durch die Tatsache, dass jedes Objekt Eigenschaften von Vaterobjekt-Typen vererben kann, lassen sich komplexe Modelle schrittweise konstruieren. Auf diese Art ist es möglich, eine Menge von High-Level-Objekttypen zu definieren, die für eine bestimmte Anwendung geeignet sind.

6.4. *Modelliertes System*

Abhängig von Einsatzgebiet.

6.5. *Modelleingabe*

Abhängig vom Einsatzgebiet.

6.6. *Modellausgabe*

Abhängig vom Einsatzgebiet.

6.7. *Schnittstellen*

Von den Autoren wurden viele High-Level-Objekte entworfen, die dieselben Modellkonstruktionen enthalten, wie sie das SAVE-Interface anbietet. Die Eingabe erfolgt in Textform, die Ausgabe mittels einer Sprache für Suchabfragen.

7. *Anwendungsfälle*

Keine bekannt.

8. *Annahmen und Einschränkungen*

Nicht bekannt.

The Mathworks: Stateflow

1. *Quellen / Referenzen*

Stateflow (2007): http://www.mathworks.com

2. Projektstatus

Die Produkte werden von der Firma The Mathworks entwickelt und sind in der Industrie als Design- und Analyse-Werkzeuge von technischen Systemen weit verbreitet. Stateflow ist neben Simulink und Matlab eines der Hauptprogramme, die von The Mathworks vertrieben

werden. Alle drei Programme sind eng miteinander verzahnt.

3. Lizenztyp

   Kommerzielle Lizenz, kostenpflichtige (ermäßigte) akademische Lizenz, kostenlose 15-Tage-Demoversion mit Betreuung (*Software Maintenance Service*).

4. Allgemeiner Zweck

   Stateflow ist ein Simulationswerkzeug für ereignisgesteuerte Systeme. Stateflow-Charts ermöglichen die graphische Darstellung von hierarchischen und parallelen Zuständen und die ereignisgesteuerten deterministischen Transitionen zwischen ihnen. Das Werkzeug erweitert herkömmliche State-Charts um folgende Konzepte: Kontrollfluss, Wahrheitstabellen, temporale Operatoren und ereignisgerichtetes Senden (*Broadcasting*). Nicht zuletzt ist Stateflow eng mit den Programmen Matlab und Simulink verknüpft, die zusätzliche graphische Aufbereitungs-möglichkeiten anbieten.

5. Plattform

   32/64-x86-PC mit Windows Vista, XP, 2000, Server sowie Linux.
   Sun-SPARC/ultraSPARC mit Solaris 8 oder höher.
   MacOS auf Intel / PPC.

6. Modell

   6.1. Modellklasse

   Quantitativ – analytisch, IT-Ebene.

   6.2. Modelltyp:

   State-Charts, endliche Zustandsautomaten (*Finite State Machines*), temporale Logik.

6.3. Modellbeschreibung

Ein Stateflow-Chart ist die graphische Repräsentation eines endlichen Zustandsautomaten (FSM), bei dem Zustände und Transitionen vom Anwender anhand von Systemblöcken beschrieben werden. Anschließend werden die Blöcke mittels Junktoren und funktionalen Beschreibungen (Wahrheitstabellen, temporale Logik) zu einem FSM vervollständigt. Es sind drei verschiedene Klassen von FSM verfügbar: Mealy- und Moore- sowie ein hybrider Mealy-Moore-Automat mit erweiterter Funktionalität. Mittels graphischer Werkzeuge werden die Automaten vom Anwender konstruiert. Mittels Sub-Charts lässt sich die Komplexität eines Stateflow-Charts reduzieren und somit in ein hierarchisch organisiertes System überführen. Die Ausführung der Stateflow-Charts erfolgt mittels Simulation.

6.4. Modelliertes System

Ereignis-gesteuerte Systeme, fehlertolerante Systeme.

6.5. Modelleingabe

Funktionale Beschreibung des Systems.

6.6. Modellausgabe

Zustand des Systems vor oder nach einem eintretenden Ereignis.

6.7. Schnittstellen

Innerhalb der Umgebung von Matlab gibt es dazu folgende Möglichkeiten:

- Definition von Eingabe/Ausgabe-Ereignissen innerhalb des Stateflow-Modells
- Verbinden von Eingabe/Ausgabe-Ports mit Simulink-Blöcken
- Definition von Simulationsparametern in Simulink

Allgemein lassen sich Schnittstellen zwischen einem Stateflow-Block und extern abgelegten Quellcode definieren. Daten über den Zustand des modellierten Systems lassen sich während der Ausführung (Simulation) aufzeichnen.

7. Anwendungsfälle

   Es gibt aufgrund der weiten Verbreitung und dem De-facto-Standard von Produkten der Fa. The Mathworks in der Industrie zahlreiche Anwendungsfälle. Auf der Webseite wird als Beispiel die Simulation eines fehlertoleranten Kraftstoff-Füllsystems vorgestellt.

8. Annahmen und Einschränkungen

   Das Werkzeug ist auf Ereignis-gesteuerte Systeme beschränkt.

2.3 Qualitative Werkzeugen

Advanced Technology Institute: OCTAVE Automated Tool

1. *Quellen / Referenzen*

   Crane et al. (2003)
   Octave (2007): http://oattool.aticorp.org/
   West, Andrew (2003)

2. *Projektstatus*

   Unbekannt. Die Webseite des Werkzeugs ist nicht verfügbar.

3. *Lizenztyp*

   Kommerzielle Lizenz (1500 US-Dollar pro Lizenz). Demo- und Testversion nicht verfügbar.

4. *Allgemeiner Zweck*

   Das Werkzeug „Octave Automated Tool“ wurde von ATI implementiert, um dem Benutzer die Anwendung des Octave- und Octave-S-Ansatzes zu erleichtern. Das Werkzeug assistiert dem Benutzer während der Datensammlungsphase, der Datenorganisation und dem Erstellen eines Abschlussberichtes.

5. *Plattform*

   Unbekannt.

6. *Modell*

   6.1. Modellklasse

Qualitativ, IT- und Prozess-Ebene.

6.2. Modelltyp

Risikoanalyse.

6.3. Modellbeschreibung

0. Phase (Vorbereitung): Sammlung von vorhandenen relevanten Materialien (z.B. Sicher-heitsrichtlinien, Architekturdiagrammen der technischen Infrastruktur, etc.), Abgleich mit dem OCTAVE-Practice-Katalog.

1. Phase (Auflistung von Aktivposten und deren Schutz): Identifizierung von kritischen, technischen und nicht-technischen Aktivposten der Organisation für jede Ebene der Organistionsarchitektur, Analyse der Bedrohungen und Herstellung einer Verbindung mit dem jeweiligen kritischen Aktivposten; Ergebnis der Phase ist eine Zusammenstellung einzelner Bedrohungsprofile.

2. Phase (Identifizierung von Beschädigungsmöglichkeiten der Infrastruktur): Bestimmung von Schlüsselkomponenten der technischen Infrastruktur, die den jeweiligen kritischen Aktivposten betreiben, und Auswahl eines Repräsentanten zu jedem Gerät für eine spätere Evaluierung; Leitung eines technischen Assessments zur Beschädigungsmöglichkeit der ausgewählten Komponente nach dem Asset-Profile-Workbook.

3. Phase (Festlegung der Auswirkung von Bedrohungen und Gegenmaßnahmen): Mittels des Asset-Profile-Workbook werden Auswirkungen von Bedrohungen, Risikoanalyse sowie Bestimmung der Anforderungen zum Schutz kritischer Aktivposten festgelegt. Basierend auf der Risikobewertung in Phase 2 werden eine Sicherheitsstrategie für das Unternehmen sowie Pläne zur Risikoverminderung festgelegt. Abschließend wird die Strategie zusammen mit dem oberen Management durchgesehen, angepasst und abgestimmt.

6.4. Modelliertes System

IT-Unternehmen.

6.5. Modelleingabe

Prozessbeschreibung.

6.6. *Modellausgabe:*

Maßnahmen, Risikobewertung.

6.7. Schnittstellen

Nicht verfügbar.

7. *Anwendungsfälle*

Auf der Webseite gibt es verschiedene Vergleichsstudien zur Implementierung der OCTAVE-Methode im Gesundheitswesen (OCTAVE - HIPAA, OCTAVE - JCAHO). Weiterhin gibt es eine Studie zur Evaluierung der OCTAVE-Methode gegenüber dem NIST-Standard SP 800-30 (*National Institute of Standards and Technology*).

8. *Annahmen und Einschränkungen*

Keine bekannt.

Alion Science and Technology Corp.: CounterMeasures

1. *Quellen / Referenzen*

CounterMeasures (2007): http://www.alionscience.com/index.cfm

2. *Projektstatus*

Alion liefert technische Expertisen und operative Unterstützung für das US-Verteidigungsministerium und weiteren Regierungsbehörden sowie für kommerzielle Kunden. Gegründet wurde Alion im Dezember 2002 aus dem bereits 1936 gegründeten IIT-Forschungsinstitut heraus. Die erste Version von CounterMeasures wurde in den 80er Jahren veröffentlicht, die letzte Version 8.1 stammt von Februar 2007.

3. *Lizenztyp*

Kommerzielle Lizenz: Es gibt drei Lizenzmodelle: „Enterprise Platform“ (US-$ 14500), „Standard Platform“ (US-$ 3990) und „Web Survey“ (US-$ 2500). Kostenlose Evaluierungslizenz. Desktop- und Netzwerkversion.

4. *Allgemeiner Zweck*

Das Softwarewerkzeug CounterMeasure ist ein Programm zur Betreibung von Risikomanagement unter anderem nach den Vorschriften der US-NIST-800-Serie und der Circular-A-130-Vorschriften des OMB (*Office of Management and Budget*). Zu den Anwendungsbereichen gehören IT-System- und Gebäuderisiken sowie operationales Risiko.

5. *Plattform*

IBM-PC oder Kompatibeles, Microsoft Windows 98/ME/NT/2000/XP, Microsoft Office 2000 oder höher.

6. *Modell*

6.1. Modellklasse

Qualitativ, Prozess-Ebene.

6.2. Modelltyp

Risikoanalyse: Identifizierung und Bewertung: Fragebögen, kundenspezifische Prüflisten, Inspektoren. Behandlung: Kosten-Nutzen-Analyse.

6.3. Modellbeschreibung

Der Benutzer standardisiert die Kriterien zur Auswertung und benutzt eine für seine Bedürfnisse zugeschnittene Prüfliste zur Bewertung. Die Software liefert objektive Auswertungskriterien, um über die Einhaltung und Erfüllung von Sicherheitsmaßstäben zu entscheiden. Im Analyseteil werden die Zusammenstellungen untersucht, die Effektivität von Gegenmaßnahmen (*Countermeasures*) bestimmt und eine Einschätzung der momentanen Beschädigungsmöglichkeiten geliefert. Darüber hinaus kann der Benutzer entscheiden, wo die größtmöglichen Risiken liegen und worin der effektivste Weg liegt, sie zu minimieren.

6.4. Modelliertes System

Physische Systeme, Prozesse.

6.5. Modelleingabe

Prozessbeschreibung.

6.6. Modellausgabe

Beschreibung von Risiken, verbunden mit den jeweiligen Abläufen und Tätigkeiten.

6.7. Schnittstellen

Die Eingabe der Daten erfolgt über eine Zusammenstellung, für die Ausgabe ist ein Reportgenerator vorhanden.

7. *Anwendungsfälle*

   Physische Sicherheit: Verkehrswege, Gebäude, Öl-und-Gas, kritische Infrastruktur.

   Informationelle Sicherheit: Best-Practice-Ansatz, OMB-A-130, NIST 800-26, HIPAA (*Health Insurance Portability and Accountability Act*), OSHA-Compliance (*Occupational Safety Health Administration*), Gap-Analysis (*Business Resource Assessment*).

   Konkrete Anwendungsfälle sind nicht verfügbar (werden jedoch auf der Webseite angekündigt).

8. *Annahmen und Einschränkungen*

   Keine bekannt.

Aprico Consultants: CASIS

1. *Quellen / Referenzen*

   Casis (2007): http://www.aprico-consult.com/

2. *Projektstatus*

   CASIS ist ein Produkt von Aprico Consultants. Informationen über das Projekt auf der Webeite sind nicht verfügbar (kein Flash-Plugin, fehlende Suchfunktion). Aktueller Status ist unbekannt.

3. *Lizenztyp:*

   Kommerzielle Lizenz, abhängig von der Funktionalität (Umfang der Logdateien, Anzahl der Anwendungsbereiche, Anzahl der Installationen, Anzahl der Seiten), ab € 45.000.

4. *Allgemeiner Zweck*

CASIS wird unter der Bezeichnung „Advanced Security Audit Trail Analyzer“ geführt. Der Zweck des Werkzeugs liegt in der Sammlung von Logdateien über verschiedene Systeme hinweg, der Korrelation dieser Daten und der Erzeugung von Sicherheitsalarmen, basierend auf benutzerdefinierten Regeln. Der Benutzer kann sowohl die Datengrundlage als auch die Ausgabe des Alarms frei bestimmen.

5. *Plattform*

Unbekannt.

6. *Modell*

6.1. *Modellklasse*

Quantitativ, IT-Ebene.

6.2. Modelltyp

Korrelationsregeln, Datentransformationen.

6.3. Modellbeschreibung

Unter anderem werden im Werkzeug Regeln zur Korrelation der Daten (*Correlation Rules*) verwendet. Dabei handelt es sich um eine abstrakte Sprache zur Beschreibung von Regeln zur Datenerhebung für das Modul „Rules Correlation Engine“ innerhalb von CASIS.

6.4. *Modelliertes System*

Unbekannt.

6.5. Modelleingabe

Gesammelte Rohdaten aus Logdateien.

6.6. Modellausgabe

Strukturierte Daten im Datenbankformat, vermutlich Daten- und Musteranalyse, Alarmgenerierung.

6.7. Schnittstellen

6.7.1. Eingabe

JAVA, XML (JAXP)-Kommandos.

6.7.2. Ausgabe

Gesammelte strukturierte Daten in Datenbanken (Oracle, SQL Server und MySQL).

7. *Anwendungsfälle*

Keine bekannt.

8. *Annahmen und Einschränkungen*

Keine bekannt.

AXIS: RA2

1. *Quellen / Referenzen*

Axis (2007): http://www.aexis.de

2. *Projektstatus*

ÆXIS ist eine Sicherheitsberatungsfirma, die im Frühjahr 1998 gegründet wurde. Seitdem ist sie an vielen verschiedenen Aktivitäten in Zusammenhang mit Informations- und IT-Sicherheit beteiligt, einschließlich Beratung und Training für Behörden, Banken, Telekommu-nikationsfirmen und andere Industriezweige. Die erste Version von RA stammt aus dem Jahre 2000. Die aktuelle Version RA2 V.1.1 ist aus dem Jahre 2005. Die Webseite ist aktuell, jedoch in einfachem HTML-Stil gehalten. Es gibt keine Screenshots und Details über die angebotenen Werkzeugfunktionen.

3. Lizenztyp

Kommerzielle Lizenz (1100 BP), Evaluierungslizenz.

4. Allgemeiner Zweck

RA2 ist ein unabhängiges Werkzeug für die Betreibung von Risikomanagement nach den Standards ISO-17799 und ISO-27001. Für jeden Schritt im Prozess bietet das Werkzeug einen entsprechenden Schritt mit anschließender Reportgenerierung der Ergebnisse an. Ein Gerät zur Datensammlung (*Information Collection Device*) kann in jedem Bereich der Organisation installiert werden, um Informationen für den Bewertungsprozess bereitzustellen. RA2 adressiert die verschiedenen Schritte im Prozess der Einrichtung und Implementierung eines ISMS (*Information Security Management System*).

5. *Plattform*

IBM-PC oder kompatibel, Microsoft Windows 95/98/ME/NT/2000/XP.

6. *Modell*

6.1. *Modellklasse*

Qualitativ, IT- und Prozess-Ebene.

6.2. *Modelltyp*

Risikoanalyse und -bewertung nach ISO/IEC 17799 und 27001.

6.3. *Modellbeschreibung*

RA2 implementiert die Prozesse zum Risikomanagement nach den Standards ISO-17799 und ISO-27001. Weitere Einzelheiten sind nicht verfügbar.

6.4. *Modelliertes System*:

IT-Organisationen.

6.5. *Modelleingabe*

Prozessbeschreibung.

6.6. *Modellausgabe*

Risikobewertung und Maßnahmenempfehlungen.

6.7. *Schnittstellen*:

Nicht bekannt.

7. *Anwendungsfälle*

RA2 wird in einigen mittelständischen Consulting-Unternehmen, unter anderem in Singapur und Japan eingesetzt. Weitere Informationen sind nicht verfügbar.

1. Annahmen und Einschränkungen

Nicht bekannt.

BMC: Remedy Suite

1. *Quellen / Referenzen*

   RemedySuite (2007): http://www.bmc.com/remedy/

2. *Projektstatus*

   Es handelt sich um ein kommerzielles Produkt, das noch unter aktiver Entwicklung steht.

3. *Lizenztyp*

   Kommerzielle Lizenz, Details sind nicht verfügbar.

4. *Allgemeiner Zweck*

   Das Werkzeug dient dazu, Verbesserungen in der Qualität des Projektmanagements über die gesamte Projektlebenszeit zu erreichen. Das Werkzeug soll ITIL-Prozesse in großen Unternehmen verwalten und automatisieren.

5. *Plattform*

   Die Softwaresuite unterstützt die Windows-Familie sowie Linux- und Unix-Betriebssysteme. Remedy ITSM kann mit Netzwerk- und Systemmanagement-Software integriert werden, wie beispielsweise BMC Patrol, HP OpenView, Tivoli Enterprise Manager. Zusätzlich kann es mit Business-kritischen Systemen, inklusive Werkzeugen zur Planung von Unternehmens-ressourcen (*Enterprise Ressource Planning*), kombiniert werden.

6. *Modell*

Remedy Software für IT-Service-Management (ITSM) ist als ITIL-kompatible Software zertifiziert worden und unterstützt zahlreiche ITIL-Prozesse:

- Incident Management
- Configuration Management
- Problem Management
- Service Level Management
- Change Management
- Availability Management

Der Hersteller behauptet, dass alle BMC-Lösungen insgesamt 27 der 34 COBIT-Kontrollziele realisieren:

- Planung und Organisation: 6 von 10 Kontrollzielen
- Akquise und Implementierung: 5 von 7 Kontrollzielen
- Auslieferung und Support: 12 von 13 Kontrollzielen
- Monitoring und Evaluation: 4 von 4 Kontrollzielen

In der Dokumentation werden keine konsistenten Definitionen der Verfügbarkeit verwendet, manchmal sind sie geschäftsorientiert (Verfügbarkeit eines Support-Centers für eine längere Zeit und geschätzte Kosten/Nutzen), in anderen Fällen wird definiert, ob ein Prozess in einem System aktiv ist (was für die Verfügbarkeit nicht in Betracht zu ziehen ist, da ein Prozess zwar aktiv sein kann aber gerade keinen Service ausführen muss) oder manchmal wird Service-Verfügbarkeit in Erwägung gezogen, aber benötigt wird eine enge Kopplung und Integration in ein System (z.B. Transaktionen unter SAP Enterprise Portal). Es wird eine proaktive Ausfallvermeidung unterstützt, jedoch sind die Methoden rudimentär entwickelt: Ein Prozess wird erneut gestartet, sobald einer seiner Parameter einen Schwellwert erreicht (beispielsweise exzessive CPU-Benutzung).

7. *Anwendungsfälle*

BMC unterstützt eine lange Liste an Firmen, die die Remedy-Suite oder Teile davon benutzen. Einige der bekannten Firmen davon sind: TeliaSonera, Infenion, Vodafone Egypt, Agfa und Dell.

8. *Detaillierte Beschreibung*

Die Architektur der Remedy ITSM-Toolsuite besteht aus einer gemeinsamen Datenbasis, einer Workflow-Engine, Möglichkeiten zur Berichterstellung und einer integrierten Entwicklungsumgebung. Die einzelnen Anwendungen innerhalb der Suite teilen sich eine gemeinsame Arbeitsplattform und ein einheitliches Datenmodell, welches den integrierten Prozessansatz innerhalb des ITIL-Frameworks unterstützt. Die Suite ist umfangreich und besteht aus 17 verschiedenen Werkzeugen, einige davon sind auch in anderen BMC-Produkten enthalten. Architektur, Organisation und Erstellungsmöglichkeiten sind ähnlich der Mercury BTO Enterprise Suite, deshalb soll nur ein kurzer Überblick über das Werkzeug gegeben werden. Remedy ITSM kann als Einzelanwendung, in Gruppen und als integrierte Anwendung betrieben werden. Einzelne Werkzeuge können standardmäßig verwendet oder kunden-spezifisch angepasst werden, um die Prozeduren und Workflows innerhalb einer IT-Organisation zu erfüllen. Von besonderer Bedeutung für die Verfügbarkeitsmessung und -bewertung ist das Remedy Service-Level-Agreement-Werkzeug, das dem Benutzer Folgendes ermöglicht:

- Planung und Verwaltung der Verfügbarkeit von individuellen und zusammengefassten Konfigurationselementen
- Überwachung von Verfügbarkeit und Leistung, um sicherzustellen, dass die Service-Level-Vereinbarungen intern und extern eingehalten werden. Die Überwachung kann kumulativ zeitbasiert, ereignisbasiert und basierend auf einem Schwellenwert erfolgen. Benutzung von eigenständigen Alarmbereitschaften, um Auslöser und Verstöße gegen das Service-Level zu identifizieren.
- Herstellung einer Relation der aktuellen IT-Serviceleistung in Bezug auf die Service-Level-Vereinbarungen.

BSI: GSTOOL

1. *Quellen / Referenzen*

   GSTOOL (1992)
   GSTOOL (2007a): http://www.bsi.bund.de/gstool
   GSTOOL (2007b): http://www.bsi.bund.de/gshb

2. *Projektstatus*

   GSTOOL ist ein vom Bundesamt für Sicherheit in der Informationstechnik (BSI) entwickeltes Softwarewerkzeug zur Unterstützung des Benutzers bei der Analyse und dem Management von IT-Risiken nach dem BSI-Grundschutz. Die aktuelle Version 4.0 stammt aus dem Jahre 2007.

3. *Lizenztyp*

   Die Abgabe des GSTOOL 4.0 an die unmittelbare Bundes-, Landes- und Kommunalverwaltung der Bundesrepublik Deutschland erfolgt kostenfrei. Für alle anderen Benutzer gibt es ein kostenpflichtiges Lizenzmodell, welches abhängig von der Anzahl der benötigen Lizenzen ist. Zusätzlich gibt es eine kostenfreie 30-Tage-Testversion mit vollem Funktionsumfang.

4. *Allgemeiner Zweck*

   Mit dem BSI-Softwarewerkzeug IT-Grundschutz (GSTOOL) stellt das BSI eine innovative und ergonomisch handhabbare Software bereit, die den Anwendern bei Erstellung, Verwaltung und Fortschreibung von IT-Sicherheitskonzepten entsprechend dem IT-Grundschutz effizient unterstützt. Nach Erfassung benötigter Informationen steht den Anwendern ein umfangreiches Berichtssystem zur Verfügung, mittels dessen er strukturierte Auswertungen über alle relevanten Daten durchführen und diese auch auf Papier oder elektronisch ausgeben kann.

5. *Plattform*

IBM-PC oder kompatibel unter Windows ME/NT/2000/XP.

6. *Modell*

6.1. Modellklasse

Qualitativ, Prozess- und IT-Ebene.

6.2. Modell

Risikoanalyse nach dem BSI-Grundschutz (BSI-Standard 100-3).

6.3. Modellbeschreibung

Die Vorgehensweise der Risikobewertung nach dem BSI-Grundschutz erfolgt zusammengefasst in folgenden Schritten:

1. Vorarbeiten:
Initiierung eines systematischen IT-Sicherheitsprozesses, IT-Strukturanalyse, Schutzbedarf-Feststellung (hinsichtlich der Grundwerte der IT-Sicherheit: Vertraulichkeit, Integrität und Verfügbarkeit), Modellierung gemäß der IT-Grundschutzkataloge und IT-Grundschutzvorgehensweise, Basis-Sicherheitscheck und ergänzende Sicherheitsanalyse.

2. Erstellung einer Gefährdungsübersicht:
Ziel ist es, eine tabellarische Übersicht über die Gefährdungen zu erstellen, die auf die betrachteten Zielobjekte des IT-Verbunds wirken. Dabei werden alle Zielobjekte gestrichen, für die kein Bedarf einer Risikoanalyse besteht. In der Gefährdungsübersicht werden die Gefährdungen pro Zielobjekt thematisch und tabellarisch sortiert. Es wird der Schutzbedarf für jedes Zielobjekt hinsichtlich der Grundwerte der IT-Sicherheit notiert.

Die Gefährdungsübersicht dient als Ausgangspunkt für die Ermittlung zusätzlicher Gefähr-dungen.

3. Ermittlung zusätzlicher Gefährdungen:
Es werden gegebenenfalls Gefährdungen des Zielobjektes berücksichtigt, die nicht im IT-Grundschutz-Modell vorgesehen sind. Dabei wird der Schutzbedarf des Zielobjekte in Relation zu den Grundwerten der IT-Sicherheit gesetzt. Relevant dafür sind Dokumentationen des Herstellers, Schwachstellen-Publikationen im Internet sowie eigene Bedrohungsanalysen.

4. Gefährdungsbewertung:
Es wird die im Schritt 2 erstellte Gefährdungsübersicht systematisch abgearbeitet und für jedes Zielobjekt und jede Gefährdung festgestellt, ob die bereits umgesetzten oder zumindest im IT-Sicherheitskonzept vorgesehenen IT-Sicherheitsmaßnahmen einen ausreichenden Schutz bieten. Die Prüfung erfolgt anhand des IT-Sicherheitskonzeptes und anhand der Kriterien Vollständigkeit, Mechanismenstärke und Zuverlässigkeit. Das Ergebnis der Prüfung wird in der Gefährdungsübersicht für jede Gefährdung einzeln in der Spalte OK vermerkt (J/N). Die Gefährdungsbewertung liefert eine Übersicht, welche Gefährdungen für die betrachteten Zielobjekte durch die Maßnahmen der IT-Grundschutz-Kataloge ausreichend Rechnung getragen ist (OK=J), und wo gegebenenfalls noch Risiken bestehen (OK=N). Die Behandlung dieser Risiken ist Gegenstand des nächsten Abschnitts.

5. Behandlung von Risiken:
In der Praxis ergeben sich im Rahmen der Gefährdungsbewertung meist mehrere Gefährdungen, denen die Maßnahmen aus den IT-Grundschutz-Katalogen nicht ausreichend entgegenwirken. Aus diesen verbleibenden Gefährdungen können sich Risiken für den Betrieb des IT-Verbundes ergeben. Deshalb muss entschieden werden, wie mit den verbleibenden Gefährdungen umgegangen wird. Für jede Gefährdung in der vervollständigten Gefährdungsübersicht mit OK=N gibt es folgende Alternativen: Risiko-Reduktion durch weitere Sicherheitsmaßnahmen, Risiko-Reduktion durch Umstruktu-rierung, Risiko-Übernahme und -Transfer.

6. Konsolidierung des IT-Sicherheitskonzepts:
Falls bei der Behandlung von verbleibenden Gefährdungen zusätzliche Maßnahmen zu den Standard-Sicherheitsmaßnahmen hinzugefügt wurden, muss das IT-Sicherheitskonzept anschließend konsolidiert werden. Dabei werden die IT-Sicherheitsmaßnahmen für jedes Zielobjekt anhand der folgenden Kriterien überprüft:
Eignung der IT-Sicherheitsmaßnahmen zur Abwehr der Gefährdungen; Zusammenwirken, Benutzerfreundlichkeit und Angemessenheit der IT-Sicherheitsmaßnahme.

6.4. *Modelliertes System*

IT-Systeme im administrativen Bereich sowie im Unternehmensbereich.

6.5. Modelleingabe

Beschreibung des IT-Systems in Textform oder tabellarisch.

6.6. Modellausgabe

Tabellarische Risikobewertung der einzelnen Zielobjekte des IT-Systems nach Vertraulichkeit, Integrität und Verfügbarkeit; Gefährdungsübersicht; Angaben von Sicherheits-maßnahmen.

6.7. Schnittstellen

Microsoft Data Access, SQL-Server; Textdateien.

7. Anwendungsfälle

Das Werkzeug wird in der Verwaltung des Bundes, der Länder und Kommunen sowie im Unternehmensbereich verwendet. Detaillierte Beschreibungen sind nicht verfügbar.

8. Annahmen und Einschränkungen

Nicht bekannt.

CALLIO: Secura 17799

1. *Quellen / Referenzen*

   Bisson, Saint-German (2007)
   Callio (2007): http://www.callio.com/

2. *Projektstatus*

   Callio Technologies wurde 2001 in Kanada gegründet und hat sich auf dem Gebiet des Managements der Informationssicherheit von Unternehmen (*Information Security Management*) spezialisiert. Callio Technologies hat es sich zum Ziel gesetzt, Unternehmen bei der Bewertung, Verwaltung und Verminderung von IT-Risiken mit Werkzeugen zu unterstützen.

3. *Lizenztyp*

   Akademische Lizenz, Demoversion.

4. *Allgemeiner Zweck*

   Das Werkzeug dient dazu, Unternehmen bei der Ausrichtung nach den Standards BS 7799 / ISO 17799 (*Information Security Management Standard*) zu unterstützen. Der Begriff Informationssicherheit umfasst folgende Bereiche: Vertrauenswürdigkeit, Integrität und Verfügbarkeit. Secura 17799 ist ein Werkzeug, mit dem ein Unternehmen ein Standard-konformes Information-Security-Management-System (ISMS) aufbauen und betreiben kann. Das ISMS liefert einen systematischen Ansatz, um sensible Informationen in einem Unternehmen zu verwalten.

5. *Plattform*

IBM-PC oder Kompatibeles, Microsoft Windows 98/ME/2000/XP, Microsoft Access 2000/2002/XP.

6. Modell

6.1. *Modellklasse*

Qualitativ, Prozess- und IT-Ebene.

6.2. *Modelltyp*

PDCA-Modell, Fragebögen.

6.3. *Modellbeschreibung*

Um die Anforderungen des Standards zu erfüllen, müssen im ISMS eine Evaluierungsmethodik und Sicherheitsrichtlinien sowie ein Dokumentations- und Review-Prozess umgesetzt sein. Das sind die grundlegenden Prinzipien des PDCA-Modells (Plan-Do-Check-Act), das wiederum dem ISO-9001-Modell für Qualitätssicherung ähnelt. Es berücksichtigt die Bereiche Angestellte, Prozesse und Informationstechnologie. Im ISMS sind die folgenden Methoden implementiert: Risikobewertung, Risikobehandlung, Methodengenerierung (*Policy Generation*) und Dokumentenmanagement. Die einzelnen Methoden werden mittels geeigneter Techniken (Fragebögen, Audits, Präsentations-vorlagen, Implementierungsvorlagen und Multimedia) unterstützt.

6.4. *Modelliertes System*

Globale Unternehmensstruktur (Angestellte, Prozesse, IT).

6.5. *Modelleingabe*

Prozessbeschreibungen, Beschreibung einzelner Aktivitäten, Risikobewertung, Ziele.

6.6. *Modellausgabe*

ISO-kompatible Diagnoseberichte, geschätztes Restrisiko, Vorschläge zur Prozessverbesserung.

6.7. *Schnittstellen*

Die Eingabe erfolgt über Multi-User-Webapplikation und für die Ausgabe existiert ein Reportgenerator.

7. *Anwendungsfälle*

Konkrete Anwendungsfälle sind nicht bekannt. Zu den Geschäftskunden von Callio zählen vor allem Firmen aus dem Bereich IT-Consulting und e-Business. In der Regel handelt es sich dabei um regional agierende Firmen (Kanada und den USA, aber auch Ungarn und Vietnam), deren erfolgreiche Geschäftstätigkeit vor allem von einem standardisierten Informationssicherheits-Management abhängig sind.

8. *Annahmen und Einschränkungen*

Nicht bekannt.

CCN-CERT: PILAR / EAR

2. *Quellen / Referenzen*

Pilar (2007): https://www.ccn-cert.cni.es

3. *Projektstatus*

PILAR wurde von der Spanish-National-Security-Agency für die spanischen Verwaltungsbehörden im Jahre 2004 entwickelt. EAR ist die freigegebene Version von PILAR für den kommerziellen Bereich. Die aktuelle Version ist Version 3.3 aus dem Jahre 2007.

4. *Lizenztyp*

PILAR darf nur innerhalb der spanischen Administration verwendet werden, von EAR gibt es eine Public-Domain-Lizenz im Read-Only-Modus und eine Benutzerlizenz für das Programm im Write-Modus. Es gibt im Write-Modus weiterhin eine Anonymous-Lizenz (Begrenzt auf 10 Aktivposten) sowie eine 30-Tage-Evaluierungslizenz des Komplettprogramms. EAR ist im Besitz von A.L.H. J. Mañas S.L.

5. *Allgemeiner Zweck*

PILAR ist ein Werkzeug, das das MAGERIT-Verfahren (*Methodology for Information Systems Risk Analysis and Management*) des spanischen Verteidigungsministeriums implementiert und erweitert. Die Funktionalität umfasst die Bereiche quantitative und qualitative Risikoanalyse sowie Risikomanagement. Weiterhin wird die Methode der Business-Impact-Analysis-and-Continuity-of-Operations unterstützt.

6. *Plattform*

IBM-PC oder kompatibel mit Microsoft Windows oder Linux.

7. *Modell*

7.1. *Modellklasse*

Qualitativ, Quantitativ-analytisch., Prozess-Ebene.

7.2. *Modelltyp*

MAGERIT-Methode mittels Tabellen sowie Attack-Trees, Prozessbeschreibungen, Audits. Quantitativ-analytische Ansätze: Datenflussdiagramme, Process-Charts, Boolesche Funktionen.

7.3. *Modellbeschreibung*

Phase 1 (Systemanalyse): Zuerst werden die Aktivposten eines Systems (Hardware, Softwareprogramme, Datenbestände, Prozesse und Personal) beschrieben, anschließend die Abhängigkeiten zwischen den Aktivposten modelliert. Die Aktivposten werden tabellarisch nach ihrer Bedeutung für das System bewertet (finanzielle Ausgaben, Verlust oder Nichtverfügbarkeit).

Phase 2 (Gefährdungsanalyse): Es werden die Gefahren analysiert (menschliche und höhere Gewalt). Relevant ist die Auswirkung einer Gefahr auf den jeweiligen Aktivposten. Auch die Gefahren werden mittels Tabellen kategorisiert und bewertet.

Phase 3 (Festlegung der Auswirkungen): Der Impact wird für jeden Aktivposten und jede Gefahr berechnet und richtet sich nach der Schwere der Gefahr und der Bedeutung des Aktivpostens für das System.

Phase 4 (Berechnung der Risiken): Es wird die Wahrscheinlichkeit einer Beschädigung des Systems durch Eintritt der Gefahr in Abhängigkeit vom jeweiligen Aktivposten berechnet.

Phase 5 (Festlegung der Restauswirkungen): Auswirkungen, die durch unvollständige Absicherungen des Systems entstehen (unwirksame Standards, Schutzvorrichtungen, Kontrollstrukturen).

Phase 6 (Berechnung des Restrisikos): Es werden unter Annahme der Restauswirkungen in Phase 5 die Wahrscheinlichkeiten der Beschädigung eines Systems berechnet.

Eine besondere Bedeutung besitzen Schutzvorrichtungen (*Safeguards*) als vorbeugende Maßnahme gegenüber Gefahren des Systems.

7.4. *Modelliertes System*

Unternehmen, Verwaltungsbehörden (gesamtheitlich).

7.5. *Modelleingabe*

Prozess- und IT-Beschreibung.

7.6. Modellausgabe

Quantitativ-analytisch: Wert und Abwertung des Aktivpostens sowie Abhängigkeit zwischen Aktivposten, Häufigkeit bedrohlicher Ereignisse, Risiko- und Restrisiko-Berechnung,

7.7. *Schnittstellen*

In Textform und graphisch.

8. *Anwendungsfälle*

PILAR wird innerhalb der spanischen Verwaltung angewendet, Anwendungen von EAR sind nicht bekannt.

9. *Annahmen und Einschränkungen*

Nicht bekannt.

C&A Systems Security Ltd.: COBRA

1. *Quellen / Referenzen*

Cobra (2007): http://www.riskworld.net

2. *Projektstatus*

Eine erste Version des Werkzeugs erschien in den 90er Jahren. Mittlerweile ist Version 3 die aktuelle Version. Es sind auf der Webseite keine Hinweise auf eine tatsächliche Existenz des Werkzeugs zu finden (Screenshots, Benutzeranwendungsfälle, etc.). Die Beschreibungen fallen knapp aus, die Webseite ist im einfachen HTML-Stil gehalten. Firmenname und Webseite stimmen nicht überein.

3. *Lizenztyp*

Kommerzielle Lizenz, COBRA Suite full/ISO17799 (US$ 1995, 895), Versuchsversion verfügbar.

4. *Allgemeiner Zweck*

Das Softwarewerkzeug COBRA ermöglicht eine Risikobewertung im Sicherheitsbereich durch das Unternehmen selbst. Es evaluiert die relative Bedeutung aller Bedrohungen und Schwachstellen, und generiert dafür die geeigneten Lösungen und Empfehlungen. Dabei werden automatisch die Risiken mit den möglichen Auswirkungen auf Geschäftsebene verbunden. Alternativ können auch einzelne Gebiete oder Angelegenheiten ohne Auswirkungen als Stand-Alone betrachtet werden. COBRA ist mit vier einzelnen Wissensbasen ausgestattet, die mittels des Modulmanagers kundenspezifisch angepasst werden können.

5. *Plattform*

Nicht bekannt.

6. *Modell*

6.1. *Modellklasse*

Qualitativ, Prozess-Ebene.

6.2. *Modelltyp*

Wissensbasierte Risikoanalyse (*Knowledge-based Risk Analysis*).

6.3. *Modellbeschreibung*

Zur Risikoanalyse des Werkzeugs gehören die Identifizierung und Bewertung eines Risikos sowie die Untersuchung möglicher Auswirkungen unter der Annahme des Eintritts eines schadhaften Ereignisses. COBRA nutzt dafür einen wissensbasierten Ansatz – in den Bereichen: IT-Sicherheit, Operational-Risk, High-Level-Risk und e-Security. Die ersten beiden Wissensbasen enthalten detaillierte Informationen zur Risikobewertung in den jeweiligen Domänen. Des Weiteren bietet die dritte Wissensbasis eine Rapid-High-Level-Bewertung für den gesamten Geschäftsbereich an. Die letzte Wissensbasis ermöglicht die Risikobewertung von modernen Netzwerk-basierten Systemen an.

6.4. *Modelliertes System*

Keine Angaben.

6.5. *Modelleingabe*

Prozessbeschreibung, Beschreibung von Bedrohungen und Schwachstellen des Systems.

6.6. *Modellausgabe*

Lösungsvorschläge und Empfehlungen, Überprüfung der ISO-17799-Einhaltung.

6.7. *Schnittstellen*

Nicht verfügbar.

7. *Anwendungsfälle*

Keine bekannt.

8. *Annahmen und Einschränkungen*

Nicht bekannt.

DCSSI: EBIOS

1. *Quellen / Referenzen*

EBIOS (2007): http://www.ssi.gouv.fr/

2. *Projektstatus*

EBIOS wird entwickelt von der französischen Central-Information-Systems-Security-Division. Die erste Version erschien im Jahre 1995 und die letzte im Jahre 2005.

3. *Lizenztyp*

Public-Domain-Software (Open-Source).

4. *Allgemeiner Zweck*

EBIOS (*Expression of Needs and Identification of Security Objectives*) ist ein Softwarewerkzeug, das die gleichnamige Methode unterstützt. Das Werkzeug erstellt die Risikoanalyse und das Management nach den fünf EBIOS-Phasen, erlaubt die Aufzeichnung aller Ergebnisse und die Erstellung einer Zusammenfassung.

5. *Plattform*

IBM-PC oder kompatibel unter Windows und Linux, PowerPC unter Linux, SPARC-Architektur unter Solaris.

6. *Modell*

6.1. Modellklasse

Qualitativ, Prozess-Ebene.

6.2. Modelltyp

EBIOS-Methode (Risikoanalyse und Bewertung).

6.3. Modellbeschreibung

Phase 1 (Untersuchung des Kontextes): Untersuchung der Unternehmens- oder Behörden-struktur, Untersuchung der IT-Systeme und Festlegung der relevanten Sicherheitsbereiche.

Phase 2 (Formulierung der Sicherheitsbedürfnisse): Bestimmung der relevanten Sicherheits-bedürfnisse und Bewertung nach Dringlichkeit.

Phase 3 (Untersuchung der Bedrohung): Aufnahme und Charakterisierung aller Angriffs-arten, Untersuchung und Einschätzung der Beschädigungsmöglichkeiten, Aus-drückliche Formalisierung und Priorisierung der Bedrohungen.

Phase 4 (Bestimmung der Sicherheitsziele): Abgleich der Bedrohungen mit den Sicherheits-bedürfnissen, Formulierung von Sicherheitszielen und Festlegung von Sicherheitsniveaus.

Phase 5 (Festlegung von Sicherheitsanforderungen): Beschreibung funktionaler

Sicherheits-anforderungen und zugesicherter Sicherheitsanforderungen mittels Richtlinien (*Level of Assurance Requirements*).

6.4. Modelliertes System

Behörden- und Unternehmensstrukturen.

6.5. Modelleingabe

Prozessbeschreibung

6.6. Modellausgabe

Tabellarische Bewertung und Einschätzung möglicher Risiken.

6.7. Schnittstellen

Graphisch und in Textform mittels Fragebögen.

7. *Anwendungsfälle*

EBIOS wird hauptsächlich im öffentlichen Bereich in Frankreich sowie in Organisationen, bei Consulting-Firmen und Firmen beliebiger Größe in der europäischen Union und Kanada eingesetzt. Nähere Angaben sind nicht verfügbar.

8. *Annahmen und Einschränkungen:*

Nicht bekannt.

Fujitsu Interstage Business Process Manager

1. *Quellen / Referenzen*

Interstage (2007): http://www.fujitsu.com/global/services/software/interstage/bpm/index.html

2. *Projektstatus*

Das Werkzeug wird als Teil der Software AG CentraSite Initiative kontinuierlich weiterentwickelt.

3. *Lizenztyp*

Eine Evaluierungsversion steht als Download zur Verfügung. Informationen zum Preis der kommerziellen Lizenz wurden bisher nicht bekannt gegeben.

4. *Allgemeiner Zweck*

Der Business Process Manager verfügt über einen Workflow-basierten Geschäfts-prozessmanager, der den gesamten Prozesszyklus umfasst: Prozessintegrierung, -automatisierung, -modellierung, -verwaltung und -optimierung. Die letzten Drei sind dabei für diese Studie interessant, da sie zu einem gewissen Grad Verfügbarkeitsmodellierung / Bewertungsmöglichkeiten bieten.

5. *Plattform*

Windows 2000, 2003, XP, Solaris 9, Red Hat Linux, ES 4.0, HP-UX 11i, IBM AIX.

Ordnerdienste: LDAP, Windows Native Directory, Microsoft Active Directory
Datenbanken: Oracle 9i, 10.2, MS SQL Server, Sybase 12.5.3, DB2 8.1

6. *Modell*

6.1. Modellklasse

Qualitativ, Service-Ebene.

6.2. Modelltyp

Workflow (BPMN, BPEL, XPDL, Wf-XML).

6.3. Modellbeschreibung

Der Business Process Manager stellt eine Umgebung zur Modellierung, zum Prüfen und zur Verfeinerung der Geschäftsprozesse und der Geschäftsregeln bereit. Diese bestimmen die Prozesse bzw. Services, bevor sie angewendet werden. Es ist ein Workflow-Modell, das unter Verwendung von Modellierungssprachen spezifiziert werden kann, wie der Business Process Modeling Notation (BPMN), der Business Process Execution Language (BPEL), der XML Process Definition Language (XPDL) oder dem WF-XML. Zusammen mit Entscheidungstabellen können Definition, Verfeinerung und Wartung von Geschäftsregeln verwirklicht werden. Die Prozesse können mittels Business Activity Monitoring (BAM) Dashboards überwacht werden. Key Performance Indicators (KPI) können definiert werden sowie Regeln, wie geantwortet wird, falls Grenzbereiche eingehalten oder überschritten werden. Neben anderen lassen sich Verfügbarkeitsmetriken in Form von Entscheidungs-tabellen definieren und als KPI konfigurieren. Die Business Process-Manager-Analytik kann Prozessanalysen und -auswertungen durch vordefinierte Metriken durchführen. Des Weiteren sind Möglichkeiten zur Suche, Sortierung und Filterung gegeben. Das Werkzeug beherrscht die Sensitivitätsanalyse (*What-If-Analysis*), womit sich zur Laufzeit Engpässe betreffend Prozesse, Leistung oder Verfügbarkeit bestimmen und eliminieren lassen. Zu diesem Zweck gibt es Audit-Trails, Echtzeit-Aktivitätsüberwachung und weitergehende Prozessanalyse.

7. Anwendungsfälle

Bank- und Finanzwesen, Energieindustrie, Reisebranche, Unternehmenssoftware und Verkehrsregelung.

8. *Annahmen und Einschränkungen*

   Verfügbarkeit wird nicht direkt unterstützt, weder während der Modellierung noch in der Analyse. Workflow-Modelle können mit Verfügbarkeitserweiterungen versehen und Metriken können individuell in Entscheidungstabellen definiert werden, was zur Laufzeit eine Verfügbarkeitsanalyse ermöglicht (durch den KPI-Mechanismus).

HP: Mercury BTO Enterprise Solutions

1. Quellen / Referenzen

   Mercury (2007): http://www.mercury.com/us/products/

2. Projektstatus

   Die Werkzeugsuite ist ein kommerzielles und aktuelles Produkt der Firma Hewlett-Packard.

3. Lizenztyp

   Kommerzielle Lizenz, die Details sind allerdings nicht bekannt. Es gibt eine Testversion, deren Benutzung auf 14 Tage begrenzt ist.

4. Allgemeiner Zweck

   Mercury BTO (*Business Technology Optimization*) Enterprise ist eine Werkzeugsuite, die den Anwender bei der Implementierung des ITIL-Service-Managements unterstützt. Dazu werden Techniken wie Dashboards, CMDB (*Configuration Management DataBase*) und eine Workflow-Engine zur Verfügung gestellt. Mercury Project and Portfolio Management bietet auch die Unterstützung von COBIT und weiteren Qualitätsstandards an. BTO besteht aus insgesamt vier Optimierungszentren und zwei Lebenszyklus-Lösungen zur Verwaltung von Applikationsveränderungen und Leistungsfähigkeit:

1. Application Change Lifecycle
2. Application Performance Lifecycle
3. Project and Portfolio Management Center
4. Quality Center
5. Performance Center
6. Business Availability Center

Die Organisation wird von der Firma zur leichteren Produktpositionierung auf dem Markt für IT-Steuerungswerkzeuge verwendet. Es gibt Überlappungen zwischen individuellen Teilsuiten.

5. Plattform

Die Software arbeitet unter der Windows-Familie sowie Linux- und Unix-Betriebssystemen. Im Einzelnen hängt die Funktionsfähigkeit von der teilweisen Integration mit anderen Softwareprodukten ab (beispielsweise J2EE Application Servers, SOA und Unternehmens-datenbanken). Es werden jedoch Produkte aller großen Entwicklungsfirmen in dieser Kategorie unterstützt.

6. Modell

Da das Ziel der Softwaresuite aus der Implementierung des ITIL-Service-Managements besteht, kann behauptet werden, dass das umfassende Modell, auf das die Software aufbaut, ITIL ist. Jedoch folgt die Software nicht streng den ITIL-Spezifizierungen und bietet auch keine klaren Grenzen zwischen den Teilen von ITIL an. Innerhalb der individuellen Werkzeuge, deren Hauptziel in der Verbesserung der Handhabbarkeit von großer Software besteht, ist das Modell oft eine einfache statistische Berechnung (z.B. Erwartungswerte) oder eine schmale Schicht, die zur Speicherung, Wiederherstellung und Visualisierung von Daten aus einer Datenbank dient.

7. Anwendungsfälle

Einige Weblinks:

- DoubleClick - http://www.doubleclick.com/ (eine der großen Firmen für das Internet-Marketing)
- Insurity - http://www.insurity.com/ (Insurity liefert Geschäftsprozess-Management für die Versicherungsindustrie)
- Navitaire - http://www.navitaire.com/ (einer der Marktführer für die Herstellung von Reservierungs- und anderen IT-Systemen)
- Grupo Posadas - http://www.posadas.com (die größte Hotelkette in Mexico und Lateinamerika).

8. Detaillierte Beschreibung

Die BTO-Suite deckt eine weiträumige Anzahl von Aktivitäten ab, die im Projektmanagement großer Unternehmen erforderlich ist (inklusive Serviceaufgaben oder Berechnung finanzieller Auswirkungen von Projektänderungen). Da dieser Bericht auf Modellierungsmethoden und -werkzeugen zur Verfügbarkeit beruht, wird die Beschreibung der Funktionalität der BTO-Suite auf Werkzeuge zur Verfügbarkeitsbewertung fokussiert.

2.3.1.1 Mercury Application Change Lifecycle

Die „Mercury Applicatioon Change Lifecycle Suite“ wird benutzt, um den Änderungsprozess einer Anwendung innerhalb der Organisation nach der Definition des Managements durchzusetzen. Das Werkzeug wird verwendet, um Veränderungen zu verwalten und das Geschäftsrisiko während des Änderungsvorganges gering zu halten. Es zielt auf eine Automatisierung des IT-Servicemanagement-Prozesses, der Reduzierung der operationalen Kosten und der Minimierung der Risiken eines Applikationsausfalls ab. Das Werkzeug unterstützt:

- Test-Management: Absicherung, dass jede Änderung die Qualitätsanforderungen unter-stützt.
- Änderungs-, Konfigurations- und Release-Management: Bewertung der Auswirkungen auf das Geschäft nach jeder Änderung, vordergründig auf Produktion und Veröffentlichung der Anwendung.
- Change-Deployment: Automatisierung des Release-Prozesses, um sicherzustellen,

dass Veränderungen erfolgreich veröffentlicht werden.

- Change-Monitoring: Alle ausgelieferten Änderungen werden auf nachteilige Beeinflussung des Geschäftsservices überwacht.

2.3.1.2 Mercury Application Performance Lifecycle and Mercury Performance Center

„Mercury Application Performance Lifecycle“ verwaltet die Leistungsfähigkeit eines allgemeinen Anwendungslebenszyklus. Es steht eng in Beziehung zum Werkzeug „Mercury Performance Center“, das eine unternehmensweite Leistungs- und Testing-Plattform realisiert. „Mercury Performance Center“ integriert sowohl eine Leistungs-Validierung als auch Diagnosemaßnahmen, während „Mercury Application Performance Lifecycle“ noch zusätzlich Folgendes unterstützt:

- Integration von „Performance Center“, „Business Availability Center“ und Werkzeugen zur Testskriptgenerierung.
- Leistungsüberwacher liefern Informationen, basierend auf der Erfahrung von Benutzern, sammeln aktuelle Fehlermeldungen und generieren Skripte für Belastungstests und Geschäftsprozessüberwachung. Leistungsprobleme können von Diagnose- und Profiler-Werkzeugen der Suite lokalisiert werden.

2.3.1.3 Mercury Project and Portfolio Management Center

„Mercury Project and Portfolio Management Center“ liefert dem Management einer Organisation auf Nachfrage Informationen über IT-Abteilungen, Projektstand und dem Status der Auslieferung von Anwendungsänderungen auf Geschäftsebene. Es ist auf die Verbesserung von Geschäftswerten ausgelegt, die von der IT erbracht werden. Es unterstützt Qualitätsprogramme und Frameworks zur Prozesskontrolle wie beispielsweise COBIT, CMMI, PRINCE2 und Six-Sigma.

2.3.1.4 Mercury Quality Center

„Mercury Quality Center“ ist ein Web-basiertes System für das Management und automatische Testen der Softwarequalität über verschiedene Entwicklungsumgebungen hinweg (z.B J2EE, .NET, Oracle, SAP). Der Zweck des Werkzeuges besteht in der Validierung der Funktionalität einer Anwendung und des automatisierten Geschäftsprozesses. Zudem kann es Engpässe in der Herstellung identifizieren.

2.3.1.5 Mercury Business Availability Center

„Mercury Business Availability Center“ ist ein Top-Down-Ansatz, um Geschäfts-, Endbenutzer- und Systemperspektiven zu integrieren und ein Bild von der komplexen Infrastruktur zu liefern, die Schlüsselanwendungen unterliegt. Es besteht aus folgenden Einzelwerkzeugen:

Mercury End User Management

„Mercury End User Management“ unterstützt die Überwachung der Verfügbarkeit von Webseiten und Anwendungen in Echtzeit. Dazu emuliert es Endbenutzer-Geschäftsprozesse und unterstützt verschiedene Protokolle in Web- und Nicht-Web-Umgebungen sowie einzelne Applikationen (z.B. Oracle, Siebel und Citrix). Das ermöglicht eine schnellere Identifizierung und Auflösung von Leistungs- und Verfügbarkeitsproblemen.

Mercury Diagnostics

„Mercury Diagnostics“ ist eine Lösung für gemischtes und traditionelles Anwendungsmanagement in Produktions- und Vorproduktionsumgebungen. Es unterstützt den Anwender bei der Lösung verschiedener Probleme, die die Geschäftsverfügbarkeit betreffen, basierend auf extrahierten Informationen von Anwendern oder scriptsynthetisierten Transaktionen. Beispiele dafür sind: Portalausfälle, Speicherüberlauf von Anwendungen und Knotenausfälle in Daten- oder Computerclustern. Es liefert Leistungsdiagnosen, womit es dementsprechend auch zum „Mercury Performance Center“ gehört. Das Werkzeug benutzt Agents, die Daten zur Leistung, Verfügbarkeit und Diagnose aus den jeweiligen Anwendungen sammeln, ohne das der Quellcode einer Anwendung oder deren Rekompilation notwendig ist. Es verwendet eine Instrumentierung des Zwischencodes (*Bytecodes*) und sammelt Systemmetriken, basierend auf Industriestandards. Die Diagnosedaten können als XML-Format exportiert werden.

Mercury Problem Isolation

„Mercury Problem Isolation“ wird verwendet, um Probleme zu identifizieren, zu diagnostizieren und zu lösen. Es kann Probleme durch die Identifizierung von Beziehungen und Abhängigkeiten innerhalb eines Systems und der Infrastruktur isolieren. Zudem

ermöglicht es den ITIL-Service-Auslieferungsprozess in Organisationen zu vervollständigen, die das ITIL-Framework benutzen. Es stellt einen Zugangspunkt für Informationen über und Lösung von Problemen dar und setzt die „Mercury Configuration Management Database“ zur Identifizierung und Darstellung von Konfigurationen und Veränderungen ein, die möglicherweise ein Problem erzeugen könnten. Es korreliert Key-Performance-Indicators (KPI) mit Konfigurationseinstellungen und verbindet die Änderungen, um das Auftreten eines möglichen Problems abzuschätzen. Wird das Werkzeug mit anderen Anwendungen des „Business Availability Centers“ (z.B. Business Process Monitor, SiteScope oder Universal CDMB) kombiniert, liefert es detaillierte Daten zu den jeweiligen problembehafteten Komponenten.

Mercury System Availability Management

„Mercury System Availability Management (SAM)“ liefert eine Lösung zur Überwachung der Infrastruktur auf der Organisationsebene. „Mercury SAM“ kann existierende Managementsysteme eines Unternehmens miteinander verbinden, um zusammen mit „Mercury Sitescope“ in der Organisation die Systemverfügbarkeit zu überwachen und Daten zur Leistung zu sammeln. SAM basiert auf einer agentenlosen Architektur und ermöglicht ein zentralisiertes Management und Konfiguration der Infrastruktur. Es kann die Informationen bezüglich der Anwendung gruppieren (z.B. CPU, Festplatten, Datenbank-Indizes und API-Werte). Zudem ermöglicht es die Verwendung als Verwaltungsanwendung verschiedener SiteScope-Montore. Die gesammelten Daten werden im SAM-Repositorium gespeichert und können aus verschiedenen Quellen kombiniert werden.

Mercury Service Level Management (SLM)

„Mercury Service Level Management “ unterstützt die Verwaltung des Service-Levels aus der Geschäftsperspektive und liefert SLA-übereinstimmende (Service Level Agreements) Berichte zu komplexen Geschäftsanwendungen in verteilten Umgebungen. Es liefert eine Abbildung zwischen den Ebenen eines Geschäftsprozesses und operationalen Anforderungen und kann im Fall eines Bruches der SLA einen Alarm ausgeben. Es liefert auch eine Brücke zwischen IT-zentrierten SLA-Metriken (CPU-Bereitschaft, Datenbankverfügbarkeit) und Metriken zur Verfügbarkeit eines Geschäftsprozesses durch Vergleich der aktuellen Leistung mit den Geschäftszielen. Weiterhin ermöglicht SLM dem Benutzer, Maßstäbe zur Verfügbarkeit und Leistung aufzustellen, die eigene Geschäftsziele

wiederzugeben, Leistung und Verfügbarkeit auf Anwenderseite zu messen und Probleme bei der Leistung und Verfügbarkeit zu isolieren, bevor SLA-Brüche auftreten.

2.3.1.6 Mercury Universal Configuration Management Database (CMDB)

„Mercury Universal CMDB " ist eine Datenbank für das Konfigurationsmanagement von IT-Unternehmen und dient zur Aufzeichnung, Dokumentierung und Speicherung von Informationen über Abhängigkeiten zwischen Geschäftsdiensten und der verbundenen Infrastruktur. CMDB kann Abhängigkeiten zwischen Konfigurationen automatisch erkennen und Konfigurationsänderungen aufzeichnen sowie Geschäftsdienste abbilden und visualisieren. Die CMDB verwaltet physikalische Konfigurationseinstellungen wie beispielsweise Server, Netzwerke und Speichergeräte als auch die zugehörigen logischen Einstellungen, wie Geschäftsanwendungen, Virtual-Private-Network, Endbenutzer und SLA. Weiterhin bietet CMDB Dienste zur Analyse von Auswirkungen, eine Zugangskontrolle und Verwaltungs-diensten zur Erzeugung und Wartung der CMDB an. Die CMDB ist eine Kernkomponente des „Mercury Business Availability Center" und des „Mercury Change Control Management" und liefert Unterstützung bei der ITIL-basierten Änderung, Konfiguration und dem Releasemanagement. Die Datenbank basiert auf den SOAP-Webservices (*Simple Object Access Protocol*) und kann Daten zwischen einzelnen Repositorien verbinden.

2.3.1.7 Mercury Application Mapping

„Mercury Application Mapping" liefert Einsichten in die dynamischen Beziehungen zwischen den Anwendungen und der darunterliegenden Infrastruktur. Es erneuert und wartet kontinuierlich diese Topologie innerhalb eines allgemeinen Beziehungsmodells, um dem Management eine Bewertung der Auswirkungen der IT auf das allgemeine Geschäft zu ermöglichen.

2.3.1.8 Mercury SiteScope

„Mercury SiteScope" ist eine agentenlose Systemüberwachungslösung, die entworfen wurde, um Verfügbarkeit und Leistung von verteilter IT-Infrastruktur (z.B. Server, Betriebssysteme, Netzwerkgeräte, Anwendungen und deren Komponenten) zu überprüfen. Der Anwender erhält Zugriff auf SiteScope über einen Webserver, um Statusinformationen einzusehen und

Konfigurationsänderungen vorzunehmen. Die Architektur besteht aus einer Anzahl von Objekten (WebPage Objects, Scheduler Objects, Monitor Objects, Alert Objects und Report Objects), die verschiedene Funktionen erfüllen. Obwohl von den Entwicklern behauptet wird, dass keine Agenten oder weitere Programme auf den Zielsystemen installiert werden müssen, ist die Formulierung teilweise irreführend, da sie einen Eingriff in das System auf unterster Ebene beinhaltet. Jedoch ist es wahr, dass keine Software zur Kommunikation der gesammelten Daten benötigt wird und dass Remote-Monitoring über das Einloggen des Benutzers auf dem System und dem Zugriff auf die Monitor-API (*Application Programming Interface*) realisiert wird. Das Überwachen des Systems und der Datentransfer der Messwerte zum SiteScope-Repositorium kosten allerdings ebenfalls Systemressourcen.

IBM Availability

Es werden die Werkzeuge „Tivoli Availability Process Manager“ und „High Availability Services“ beschrieben.

2.3.1.9 IBM Tivoli Availability Process Manager

1. *Quellen/Referenz*

   Tivoli (2007): http://www-306.ibm.com/software/tivoli/products/availability-process-mgr/

2. *Projektstatus*

   Das Werkzeug wird stetig weiterentwickelt.

3. *Lizenztyp*

   Kommerzielle Lizenz. Die Testversion steht als Download zur Verfügung.

4. *Allgemeiner Zweck*

Der Tivoli Availability Process Manager erlaubt die Verfügbarkeitsbestimmung und die Prioritätszuweisung bei Vorfällen entsprechend ihrer Auswirkungen sowohl auf die IT-Infrastruktur als auch auf kritische Geschäftsprozesse. Auf diese Weise erzeugt es eine Verbindung zwischen der IT- und der Geschäftsebene. Es bietet auf ITIL basierende Verfügbarkeitsverwaltung bzw. Verwaltung von Vorfällen und bestimmt die Auswirkung eines Ausfalls auf das Geschäft.

5. *Plattform*

IBM AIX 5.2,5.3, RedHat Linux, Microsoft Windows.

6. *Modell*

6.1. Modellklasse

Qualitativ, IT- und Service-Ebene.

6.2. *Modelltypen*:

Component Failure Impact Analysis (CFIA) und Verfügbarkeitsverwaltung (Verwaltung von Vorfällen), wie durch ITIL spezifiziert.

6.3. *Modellbeschreibung*:

Das Modell bietet zwei grundlegende Fähigkeiten: Verfügbarkeitsverwaltung und die Feststellung geschäftlicher Auswirkungen von Ressourcen und Vorfällen auf den Geschäftsprozess.

Verfügbarkeitsverwaltung

Die Information Technology Infrastructure Library (ITIL) definiert das Ziel der Verfügbarkeitsverwaltung als Optimierung des Leistungsvermögens der IT-Infrastruktur und Unterstützung der Organisation. Dadurch wird ein kostengünstiges und anhaltendes

Level der Serviceverfügbarkeit erreicht, der dem Geschäft erlaubt, seine Vorgaben zu erfüllen. Die ITIL-Definition von Verfügbarkeitsverwaltung betrifft nur die Verfügbarkeit der Service-Ebene und überlässt die Verfügbarkeit der Infrastruktur-Ebene einer Menge von optimalen Verfahren, die unter *Service-Betrieb* behandelt werden. Der Tivoli Availability Process Manager erweitert diesen Prozess durch Einbeziehung verschiedener ITIL-Prozesse, welche die ITIL-Service-Delivery und den ITIL-Service-Support umfassen. Diese Prozesse sind unter anderem:

- Ereignisverwaltung
- Verwaltung von Vorfällen
- Problem-Management
- Verwaltung der Service-Ebene
- Verfügbarkeitsverwaltung

Dieses Werkzeug stellt Funktionen zur Verfügung, die dabei helfen, geschäftsbezogene Auswirkungen von Vorfällen oder Serviceunterbrechungen zu bestimmen und zu verstehen. Weiterhin liefert es kritische Informationen, welche die Klassifizierung und Priorisierung von Vorfällen oder Serviceunterbrechungen und den Umgang damit erleichtern. Das Ziel der Verwaltung von Vorfällen ist in ITIL die Minimierung der Geschäftsunterbrechung durch schnellstmögliche Wiederherstellung eines vereinbarten Service-Betriebs-Levels. Der ITIL-Prozess zur Verwaltung von Vorfällen ist in sieben Schlüsselbereiche aufgeteilt:

1. Vorfallserkennung
2. Vorfallsklassifizierung
3. Erste Maßnahmen
4. Untersuchung und Diagnose
5. Lösung und Wiederherstellung
6. Vorfallsabschluss
7. Vorfallsbeobachtung und -kommunikation

Der Schwerpunkt des Tools liegt auf Erkennung, Klassifizierung und ersten Maßnahmen sowie Untersuchung und Diagnose von Vorfällen, weiterhin auf der Feststellung der

Auswirkungen auf das Geschäft.

Der Tivoli Availability Process Manager enthält die Funktion Determine Business Impact (DBI), welche Nutzer bei der Einschätzung und Priorisierung der Auswirkungen unterstützt, die verschiedene Ressourcen auf das Geschäft haben. Die DBI-Funktion hilft bei der Priorisierung eines Vorfalls anhand der Ressource, für die ein Problem gemeldet wird. Beeinflusst wird diese Entscheidung durch die betroffenen Service Level Agreements (SLA) sowie den aktuellen Zustand der Ressource, potentiell ausfallender Komponenten, betroffennr Geschäftssysteme und anderen Gruppierungen. Mit dieser Funktion lässt sich auch die Auswirkung von Änderungen an Ressourcen einschätzen. Dabei ist der aktuelle Zustand nicht so wichtig wie die Beziehungen zu Geschäftssystemen und anderen Gruppierungen als auch die SLAs, die für zu ändernde Ressourcen gelten. Das Werkzeug erhält Informationen über IT-Beziehungen aus der Configuration Management Database. Folgende Schritte sind in der DBI-Funktion enthalten: Suche, Einschätzung ausfallender Komponenten, Einschätzung der betroffenen Services, Einschätzung der Auswirkungen auf SLA/OLA und eine Zusammenfassung.

7. *Anwendungsfälle*

Automobilindustrie, Bankwesen, Computer-Dienstleister, Energiewirtschaft, Behörden, Gesundheitswesen, Medien.

8. *Annahmen und Einschränkungen*

Das Werkzeug kann nur im Tivoli Unified Process (TUP) sinnvoll und effizient genutzt werden, der allerdings eine Hersteller-Vereinbarung mit IBM verlangt.

2.3.1.10 IBM High Availability Services

1. *Quellen / Referenzen*

HAS (2007): http://www-935.ibm.com/services/us/index.wss/offerfamily/bcrs/a1026936

2. *Projektstatus*

   Hierbei handelt es sich um einen von IBM angebotenen Beratungsservice.

3. *Lizenztyp*

   Nicht verfügbar. Eine Beratung muss mit IBM vereinbart werden.

4. *Allgemeiner Zweck*

   IBM High Availability Services helfen bei der Vermeidung teurer Ausfälle und Wiederherstellungen. Sie unterstützen Bestimmung, Planung, Entwurf, Konstruktion, Implementierung und Verwaltung einer Infrastruktur, welche dauerhafte Geschäftsverfügbarkeit und Einhaltung von Service-Level-Vorgaben ermöglicht. Weiterhin soll so eine einheitliche Verwaltung von kritischen Geschäftsprozessen, IT-Systemen, Betriebsumgebungen und Netzwerken erreicht werden. Durch Nutzung dieser Services können Ausfallzeiten reduziert, die Komplexität der Infrastruktur verringert und die Nutzung von IT-Ressourcen verbessert werden. Folgende Service (Consulting)-Optionen werden angeboten:

   - High Availability: Bestimmung
     - Analyse von IT-Verfügbarkeitsplänen, -Prozessen, -Abläufen, -Funktionen, -Zuständigkeiten, -Berichten und -Steuerungen sowie das Erreichen eines Service-Levels.
     - Analyse von Ausfällen, die nach einer Vorfallsprüfung auftreten könnten, Kosten eines Ausfalls oder von CFIA-Techniken (*Component Failure Impact Analysis*).

   - High Availability: Planung und Entwurf
     - Planung für High Availability, dazu gehören Pläne, Programm-Verwaltung, Berichte, und Service-Level-Verwaltung
     - Prozess- und Ablaufentwürfe für High Availability, Funktionen und Zuständigkeiten eingeschlossen

- Architekturentwürfe für High Availability-Technologie, die IBM-Systeme z, i, p und x sowie Server anderer Hersteller als auch für Speicher und Netzwerke

- High Availability: Implementierung
  - Planung, Entwurf und Implementierung der IBM Serveroptimierungs- und Integrationsservices – umfasst: IBM Implementation Services for Geographically Dispersed Open Clusters, IBM Geographically Dispersed Parallel Sysplex (GDPS) Technologien, die Betriebssysteme IBM AIX und Microsoft Windows

- High Availability: Leistungsservice
  - Planmäßiger Leistungsservice für die IBM-Systeme z, i und p sowie Server anderer Hersteller; Speicher von IBM und anderen Herstellern als auch Netzwerke
  - Betrieb von Rechenzentren und Sicherung der Arbeitsplatzkontinuität

5. *Plattform*

Nicht verfügbar.

6. *Modell*

6.1. Modellklasse

Qualitativ, Service-Ebene.

6.2. Modelltypen

Fragebogen.

6.3. Modellbeschreibung

Die Bestimmung der Verfügbarkeit erfolgt hier durch eine qualitative

Fragebogenmethode, die geschäfts-, daten- und ereignisbedingte Gefahren identifiziert. Im Folgenden sind einige Fragebogenbeispiele aus allen 3 Bereichen aufgelistet:

Geschäftsbedingte Gefahren

Dies beinhaltet Geschäftskontinuitätsverwaltung und -regeleinhaltung. Beispielfragen:

- Wie werden die an Regierung und Industrie gerichteten Anordnungen lokal und quer über den Globus geregelt?
- Inwiefern wurden Vorbereitungen getroffen, die Risiken zu verringern, falls eine Nicht-Einhaltung vorliegt?
- Inwieweit richtet sich das Geschäft auf Erfolg ein?

Datenbedingte Gefahren

Dies beinhaltet die Beschädigung, den Diebstahl und den Verlust von Daten. Beispielfragen:

- Sind die Sicherungskopien der Daten an einem gefährdeten Ort gespeichert?
- Wurde der Wiederherstellungsvorgang getestet, damit sicher gegangen werden kann,
  dass er funktioniert?
- Gibt es einen Notfallplan, falls die Sicherungskopien gelöscht wurden?
- Können Daten auf Anforderung während einer Prüfsituation erzeugt werden?
- Können autorisierte Benutzer innerhalb eines vorgegebenen Zeitfensters auf die Daten zugreifen?

Ereignisbedingte Gefahren

Diese Ereignisse können nicht vorhergesagt werden, allerdings können Vorbereitungen getroffen werden, die im Ernstfall helfen. Solche Gefahren reichen von normalen Unfällen bis zu terroristischen Akten und zu Pandemien. Beispielfragen:

- Wie werden Angestellte behandelt, die nicht arbeiten können oder wollen?
- Was geschieht, falls keine normale Kommunikation möglich ist?
- Was geschieht bei eingeschränktem Luftverkehr, rationiertem Kraftstoff oder einer begrenzten Anzahl an Mietwagen?

7. *Anwendungsfälle*

   Keine bekannt.

8. *Annahmen und Einschränkungen*

   Nicht bekann.

Information Governance: PROTEUS

1. *Quellen / Referenzen*

   Proteus (2007): http://www.infogov.co.uk

2. *Projektstatus*

   Im Jahre 1995 wurde PROTEUS vom BS (*British Standards Institution*) offiziell als Werkzeug zur Automatisierung der Norm BS-7799 (*Code of Practice for Information Security Management*) empfohlen. Die aktuelle Version stammt aus dem Jahre 2007.

3. *Lizenztyp*

   Kommerzielle Lizenz (PROTEUS Solo 599 BP/Jahr, PROTEUS Professional 6000 BP/Jahr), WebEx-Demo auf Nachfrage.

4. *Allgemeiner Zweck*

PROTEUS ist ein Webserver-basiertes Werkzeug zur Unterstützung des Informationssicherheits- und Risikomanagements sowie zur Überprüfung der Einhaltung von Standards (ISO/IEC 17799 und BS ISO/IEC 27001, BS 25999, COBIT, PCI DSS, etc.).

5. *Plattform*

IBM-PC oder Kompatibeles.
Client: Microsoft Windows NT/2000/XP.
Server: Microsoft Windows Server 2003, Linux.

6. *Modell*

6.1. Modellklasse

Qualitativ, IT- und Prozess-Ebene

6.2. Modelltyp

Risikoanalyse und -bewertung.

6.3. Modellbeschreibung

PROTEUS bietet einen mehrstufigen generischen Prozess nach BS ISO 27001 oder einer anderen Methodik an:

- Risiko-Identifizierung: Qualitative und quantitative Risikobewertung (Aktivposten-Management, Bedrohungen, Gegenmaßnahmen, Pläne zur Risikobehandlung und Unfallmanagement)

- Risiko-Analyse: Relative und absolute Risiko-Skalen

- Risiko-Auswertung: Die Auswertung erfolgt auf fünf Ebenen: Technik, Information, Dienste, Anwendungen und Gruppen (Kombinationen).

Bedrohungen können aus Beziehungen zwischen den Aktivposten abgeleitet werden

6.4. *Modelliertes* System

IT-Unternehmen.

6.5. Modelleingabe

Prozessbeschreibung.

6.6. Modellausgabe

Risiko-Skalen, Maßnahmenempfehlungen.

6.7. *Schnittstellen*

Datenaustausch mittels XML-Format (Import: Liste der Bedrohungen, Gegenmaßnahmen, Aktivposten, Organisationsstrukturen, Betriebsunterbrechungen, Benutzer, Geschäfts-einheiten, Projekte), graphische Ausgabe der Analyseergebnisse und Maßnahmen.

7. *Anwendungsfälle*:

Keine bekannt.

8. Annahmen und Einschränkungen

Nicht bekannt.

1. *Quellen / Referenzen*

   CRAMM (2007): http://www.cramm.com

2. *Projektstatus*

   Das Werkzeug CRAMM (*CCTA Risk Assessment and Management Method*) wurde erstmalig 1985 von der britischen CCTA (*Central Computer and Telecommunications Agency*) als Auftragsarbeit für britische Regierungsbehörden veröffentlicht. Nach dem Erfolg der ersten Version wurde 1988 eine weitere Version für den Einsatz im kommerziellen Bereich entwickelt. Version 5.1 aus dem Jahre 2005 mit Anpassungen an die aktuelle BS-7799-Norm ist die letzte veröffentlichte Version.

3. *Lizenztyp*

   Kommerzielle Lizenz, 30-Tage-Evaluierungsversion.

4. *Allgemeiner Zweck*

   CRAMM ist ein Softwarewerkzeug zur Automatisierung des CRAMM-Prozesses. Ursprünglich wurden die existierenden Methoden zur Risikoanalyse und -bewertung der britischen Behörde für Informationssicherheit (*Central Government for Information Security*) in dem Werkzeug umgesetzt. Die Methode CRAMM liefert einen stufenbasierten Ansatz, der sowohl technische (IT-Hardware und -Software) als auch nicht-technische Aspekte (Mensch, Physis) enthält: Identifizierung eines Aktivpostens und dessen Einschätzung, Bedrohungs- und Schadensanfälligkeitsbewertung sowie eine Auswahl von Gegenmaßnahmen und Empfehlungen.

5. *Plattform*

   Nicht bekannt, vermutlich IBM-PC oder kompatibel mit Microsoft Windows.

6. *Modell*

6.1. *Modellklasse*

Qualitativ, Prozess-Ebene.

6.2. *Modelltyp*

Risikoanalyse und -bewertung nach CRAMM.

6.3. *Modellbeschreibung*

Die Analyse eines Risiko vollzieht sich in drei Stufen:

1. Aktivpostenbewertung und Einschätzung: Der Reviewer ermittelt und kategorisiert die Aktivposten in Hardware, Software, Daten und lokale Aktivposten. Jeder Aktivposten kann eine Einschätzung erhalten: technische physikalische Aktivposten durch ihre Ersatzkosten. Daten und Software werden nach den Auswirkungen bewertet, die durch ihren Verlust oder ihre Nichtverfügbarkeit entstehen.

2. Bedrohungs- und Schadensanfälligkeitsbewertung: Es wird geschätzt, wie wahrscheinlich das Auftreten potentieller Probleme bei den Aktivposten ist. Als Bedrohung werden verschiedene Szenarien aufgeführt (Hacking, Virus, Ausfall von Geräten oder Software, mutwillige Zerstörung, Terrorismus oder menschliches Fehlverhalten). Die Stufe wird mit einer Berechnung des Risikoniveaus abgeschlossen.

3. Auswahl an Gegenmaßnahmen und Empfehlungen: CRAMM besteht aus über 3000 detaillierten Gegenmaßnahmen organisiert in 70 logischen Gruppen. Das CRAMM-Werkzeug vergleicht das gemessene Risikoniveau der vorangegangenen Stufe mit dem Sicherheitslevel der jeweiligen Gegenmaßnahme, um zu entscheiden, ob das Risiko hoch genug ist, diese Gegenmaßnahme zu rechtfertigen.

6.4. *Modelliertes System*

Nicht bekannt.

6.5. *Modelleingabe*

Prozessbeschreibung.

6.6. *Modellausgabe*

Risikobewertung sowie Gegenmaßnahmen und Empfehlungen.

6.7. *Schnittstellen*

Eingabe von Review-Dokumenten, die Ausgabe erfolgt mittels Microsoft Word, Excel oder Graph.

7. *Anwendungsfälle*

CRAMM wird benutzt unter anderem von BAE Systems, IBM, General Motors, Royal Air Force, Swiss Bank und T-Mobile. Näher beschriebene Anwendungsfälle sind nicht bekannt.

8. *Annahmen und Einschränkungen*

Nicht bekannt.

RiskWatch: RiskWatch for Information Systems

1. *Quellen / Referenzen*

RiskWatch (2007): http://www.riskwatch.com

2. *Projektstatus*

RiskWatch liefert Software zur Sicherheitsanalyse, die es Unternehmen ermöglicht, Risiko-basierte Assessments auszuführen und ihre Organisation an den ISO-Standards zur Informationssicherheit auszurichten. Auf der weiterhin aktuellen Webseite sind keine Angaben zur Softwareentwicklung verfügbar.

3. *Lizenztyp*

Kommerzielles Lizenzmodell. Online-Repräsentation.

4. *Allgemeiner Zweck*

Das Werkzeug „RiskWatch for Information Systems“ dient zur automatischen Risikoanalyse und Bewertung der Schadensanfälligkeit von Informationssystemen. Die mitgelieferte Wissensbasis ist kundenspezifisch anpassbar. Es lassen sich neue Kategorien von Aktivposten, Bedrohungen, Schadensanfälligkeiten, Sicherheitsbarrieren und Fragebögen definieren.

5. *Plattform*

IBM-PC oder kompatibel, Microsoft Windows XP und Office Professional.

6. *Modell*

6.1. Modellklasse

Qualitativ, IT- und Prozess-Ebene.

6.2. Modelltyp

Risikoanalyse und -bewertung.

6.3. Modellbeschreibung

Das Werkzeug enthält weiterhin Implementierungen des ISO-17799-Standards, des US-NIST-800-26-Standards, nach COBIT-4.0 und Sarbanes-Oxley (SOX). Nähere Einzelheiten sind nicht verfügbar.

6.4. Modelliertes System

IT-Systeme.

6.5. Modelleingabe

Prozessbeschreibung.

6.6. Modellausgabe

Maßnahmenempfehlungen zur Erhöhung der Sicherheit nach den beschriebenen Standards.

6.7. Schnittstellen

Keine bekannt.

7. Anwendungsfälle

Für das Werkzeug „RiskWatch Information Systems“ sind keine Anwendungsfallstudien auf der Webseite verfügbar.

8. Annahmen und Einschränkungen

Keine bekannt.

Self-assesements Programs by itSMF International

1. *Quellen / Referenzen*

   itSMF (2007): http://www.itsmf.com/bestpractice/selfassessment.asp

2. *Projektstatus*

   Das Projekt ist aktiv.

3. *Lizenztyp*

   Das Werkzeug ist frei verfügbar entweder als Stapel von Excel-Spreadsheets oder online.

4. *Allgemeiner Zweck*

   Der Zweck des Fragebogens ist es, der Organisation (und ihrem Management) eine Idee zu geben, wie gut der derzeitige Stand im Vergleich zum ITIL-Best-Practice-Ansatz ist. Der Fragebogen kann nicht zu Testzwecken derart, ob es eine komplette Übereinstimmung mit ITIL gibt, verwendet werden. Jedoch sollte er das Bewusstsein beim Management erzeugen, was getan werden kann, um die Leistungsfähigkeit des Gesamtprozesses zu verbessern.

5. *Plattform*

   Online, jede Plattform die Microsoft Excel unterstützt.

6. *Modell*

   Das Schema zur Selbstbewertung besteht aus einem einfachen Fragebogen, der es dem Benutzer ermöglicht festzustellen, welche Gebiete bearbeitet werden müssen, um die Leistungsfähigkeit des Gesamtprozesses zu verbessern. Die Fragen sind vom Ja/Nein-Typ verbunden mit Gewichten bei den Ja-Antworten (Nein entspricht einer Null). Auf Basis der

Antworten wird entschieden, ob eine Organisation eine Handlungsanweisung (*Guideline*) in einer bestimmten Kategorie implementieren sollte. Um den Test innerhalb des Niveaus einer bestimmten Kategorie zu bestehen, müssen alle Pflichtfragen mit Ja beantwortet werden. Zusätzlich werden die Gewichte der nicht-Pflichtfragen aufsummiert, wobei höhere Werte besser sind. Die Ergebnisse der Selbst-Evaluierung können zur statistischen Analyse an das itSMF gesendet werden. Die Analyse wird anonym vollzogen, und es ist möglich, die gemittelten Resultate über alle bisher teilgenommenen Organisationen zu betrachten. Das ist sinnvoll für den Vergleich erhaltener Resultate mit Ergebnissen anderer Organisationen.

7. *Anwendungsfälle*

Die Analyseergebnisse, die auf der Webseite verfügbar sind, werden anonymisiert, so dass keine Namen von Teilnehmern bekannt sind. Jedoch kommen die meisten von ihnen aus dem IT-Consultant-Sektor (24.2 %), gefolgt von Regierungsorganisationen (18%). Nicht jede Organisation nimmt an jedem Fragebogen teil, so dass die Gesamtzahl variiert. Bei der Bewertung des Verfügbarkeitsmanagements haben insgesamt 677 Organisationen das Werkzeug verwendet sowie beim Service-Level-Management beeindruckende 7415 Organisationen.

8. *Detaillierte Beschreibung*

Für jede ITIL-Kategorie (Service Level Management, Financial Management, Capacity Management, Continuity Management, Availability Management, Service Desk, Incident Management, Problem Management, Configuration Management, Change Management und Release Management) werden die Fragen auf 10 Ebenen (Levels) gruppiert:

Level 1 (Prerequisites): Es wird bestimmt, ob ein Minimum an Mitteln vorhanden ist, um die Prozessaktivität zu unterstützen.

Level 1.5 (Management Intent): Das Level ist etabliert, falls es organisatorische Methoden und Anweisungen sowie Geschäftsziele oder Handlungsabsichten gibt, die Zweck und Führung oder beides in der Umgestaltung oder der Benutzung von Vorbereitungen anbieten.

Level 2 (Process Capability) Es werden die ausgeführten Aktivitäten untersucht. Die Fragen richten sich danach, ob eine minimale Anzahl an Aktivitäten ausgeführt wird.

Level 2.5 (Internal Integration): Es wird bestimmt, ob die Aktivitäten zur Erfüllung der Prozessabsichten erfolgreich integriert wurden.

Level 3 (Products): Untersucht wird der aktuelle Ausstoß des Prozesses, um festzustellen, ob alle notwendigen Produkte produziert werden.

Level 3.5 (Qualitätskontrolle): Qualitätskontrolle beschäftigt sich mit der Durchsicht und Verifizierung des Prozessaustoßes, um eine Übereinstimmung mit der beabsichtigten Qualität sicherzustellen.

Level 4 (Management Information): Management-Information beschäftigt sich mit der Leitung eines Prozesses und stellt sicher, dass adäquate und pünktliche Informationen vom Prozess bereitgestellt werden, um notwendige Entscheidungen des Managements zu unterstützen.

Level 4.5 (External Integration): External-Integration bestimmt, ob alle externen Schnittstellen und Beziehungen zwischen einzelnen Prozessen innerhalb einer Organisation eingerichtet sind. Auf dieser Ebene wird vom IT-Servicemanagement eine komplette Benutzung der ITIL-Terminologie erwartet.

Level 5 (Customer Interface): Das Customer-Interface beschäftigt sich hauptsächlich mit laufenden externen Reviews und der Validierung des Prozesses, um sicherzustellen, dass Kundenwünsche optimal behandelt werden.

Software AG: CentraSite

1. *Quellen / Referenzen*

CentraSite (2007): http://www.softwareag.com/Corporate/products/centrasite/default.asp

2. *Projektstatus*

   Es ist ein kommerzielles Werkzeug,, das kontinuierlich weiterentwickelt wird.

3. *Lizenztyp*

   Eine Evaluierungsversion steht als Download zur Verfügung. Informationen zum Preis der kommerziellen Lizenz wurden bisher nicht bekannt gegeben.

4. *Allgemeiner Zweck*

   CentraSite ist eine Plattform zur Betreibung einer Administration auf SOA-Basis (*Service-oriented Architecture Governance*). Diese umfasst zwei Aspekte: Service Implementation und IT-Governance auf Geschäftsprozess-Ebene. Die Governance-Prioritäten liegen bei der Leistung, dem Risikomanagement, der Service-Verfügbarkeit und dem Ausrichten der IT-Infrastruktur nach den Geschäftszielen (Prozesse). CentraSite unterstützt die Administrierung durch einen schrittweisen Prozess. Dieser beinhaltet die Kontrolle sowie Verwaltung von Service-Ebenen und Verfügbarkeit, die Beobachtung der Einhaltung der Service-Policys sowie die Optimierung von Prozessen, die mittels Services erstellt wurden (und vice versa).

5. *Plattform*

   Plattformunabhängig (Java- sowie Browser-basierte graphische Benutzeroberfläche).

6. *Modell*

   6.1. Modellklasse

   Qualitativ, Service-Ebene.

   6.2. Modelltyp

Metadaten-Repositorium, das Policys unterstützt; Änderungssteuerung, Wartung, Automatisierung, Abhängigkeitsanalyse und Auswertung auf Geschäftsprozess- bzw. Service-Ebene.

6.3. Modellbeschreibung

Der Zweck des SOA-Governance-Modells ist eine auf den Geschäftszielen basierende Ausrichtung von IT-Vorgängen, -Prioritäten und -Entscheidungen. Es soll dafür sorgen, dass die IT mit einer hohen Effektivität zur Unterstützung der Geschäftziele betrieben wird. Das Governance-Modell umfasst die Anwendung von Policys zur Risikoredzierung, die Umsetzung von konsistenten Prozessen und das Einrichten allgemeiner Metriken zur Leistungsüberwachung und zur Service- und Verlässlichkeitsqualität. Im Rahmen dieser Studie ist die letzte Eigenschaft von hoher Bedeutung. Beispielsweise muss ein erfolgreiches Governance-Modell garantieren, dass durch Services die Leistung oder Verfügbarkeit eines internen Back-End-Systems (wirksame Verbindung von Geschäts-prozess- bzw. Service- und IT-Ebene) nicht gefährdet werden. Das SOA-Governance-Modell involviert Geschäftsservices über den gesamten Lebenszyklus des Services, vom Entwurf bis zur Laufzeit. Die Governance der Entwurfsphase legt die Grundregeln zur Erzeugung eines Services, die architektonischen Standards und die Wartungsverträge fest. Bei der Laufzeit-Governance wird auf Übereinstimmung mit dem Vertrag und den Policys, der Leistung, Service Level Agreements und QoS (z.B. Verfügbarkeit, Verlässlichkeit und Sicherheit) geprüft. Zudem beinhaltet das Modell Veränderungsphasen, die die Durchführung von Meta-Policys sowie die Ausführung einer Auswirkungsanalyse von Serviceveränderungen bei IT-Funktionen, System- / Datenschnittstellen und Vertragsüber-einstimmung erfordern.

CentraSite beruht auf dem Ablage-Governance-Modell, bei dem die Ablage Metadaten über die Services speichert, um das Schichtenmodell zwischen Services und Implementationen bereitzustellen. Im Detail bedeutet das, dass die Ablage Modelle, Abbildungen, Shared-Keys, Transformationen, Prozessmodelle, Geschätsregeln, Testpläne, Laufzeit-Leistungs-darstellungen und andere Dinge speichert.

CentraSite beinhaltet mehrere Werkzeuge, von denen einige in Hinsicht auf die Ziele dieser Studie relevant sind. Jene werden ausführlich beschrieben, während andere auf Grund der Vollständigkeit ausschließlich erwähnt werden. Der Fujitsu Interstage Business Process Manager wird gesondert betrachtet.

Crossvision Information Integrator ist ein Werkzeug zur Organisation, Verwaltung, Ansammlung, Transformation und Filterung von Geschäftsdaten. Als solches ist es folglich nicht für die aktuelle Studie von Bedeutung.

Crossvision Service Orchestrator eignet sich für das Routing von Nachrichten, die Schema Validierung, die Tranformation und die Sequenz-Workflows, wobei alle auf XML basieren, und für Webdienste. Als solches eignet sich auch dieses nicht für die aktuelle Studie.

Ilog verwaltet Geschäftsentscheidungen durch regelbasierende Entscheidungsservices. Dabei ist für diese Studie interessant, dass die Regeln auf QoS-Metriken basieren können (wie z.B. Verfügbarkeit).

Mega Tool stellt Auswertungen und analytische Informationen während der Zuordnung von Services und Geschäftszielen zur Verfügung. Es ist möglich die Anpassung der ausgewählten Dienste zu den Geschäftsprozessen der Organisierung zu messen oder die Verfügbarkeits- und Abhängigkeitsebenen zwischen Services und Geschäftsprozessen abzutasten.

Parasoft Tool unterstützt wiederverwendbare Tests, Regressionstests, emulierte Services, Funktions-, Leistungs- und Verfügbarkeitstests.

AmberPoint Tool sichert zur Laufzeit die Einhaltung von Vorgaben, die in der Entwurfsphase gemacht wurden. Es kann die folgenden Sonderfälle feststellen: Abhängigkeiten, fehlerbehaftete Dienste, dynamische Änderungen am Service-Netzwerk, Service-Ebenen, Sicherheit, Revision, Protokollierung, Debugging und Ausfälle. Policys dienen der Kontrolle des Laufzeitverhaltens, um hohe Verfügbarkeit, Leistung und Sicherheit zu gewährleisten. Diese werden automatisch durchgesetzt.

7. *Anwendungsfälle*

   Alle Gemeinschaftsmitglieder, darunter Fujitsu, Novell und Software AG, verwenden CentraSite.
   Unter http://www.softwareag.com/Corporate/Solutions/Customers/References/default.asp sind fertiggestellte Projekte zu finden, die CentraSite nutzen. Bekannte Nutzer sind beispielsweise Belgian National Railway Company, Commerzbank, DaimlerChrysler, Nissan Motors, Scandinavian Airlines, Vodafone, Volkswagen und ZDF.

8. *Annahmen und Einschränkungen*

   Das Werkzeug basiert ausschließlich auf Webdiensten (SOAP, WSDL, UDDI). Zudem bietet es ein proprietäres Metadatenmodell, Verfügbarkeitspolicys und Metriken, die schwer integrierbar sein könnten.

Telindus Consultants Enterprises: ISAMM

1. *Quellen / Referenzen*

   ISAMM (2007): http://www.telindus.com

2. *Projektstatus*

   Das Werkzeug wurde zur internen Verwendung bei Telindus im Jahre 2002 entwickelt.

3. *Lizenztyp*

   Unbekannt, Werkzeug ist nicht verfügbar.

4. *Allgemeiner Zweck*

ISAMM ist ein Werkzeug zur Betreibung von Risikomanagement. Es berechnet einen idealen Sicherheits-Sanierungsplan, der alle bedeutsamen Aktionen zur Erhöhung der Sicherheit nach der Wirtschaftlichkeit – dem ROSI-Wert (*Return On Security Investment*) – berechnet. ISAMM Risikobewertung bedeutet für den Kunden vorzugsweise Beratungsdienste (*Consultancy Services*) und geführte Assessments nach der ISAMM-Methodik (*Information Security and Monitoring Method*). Weiterhin werden in Verbindung mit ISAMM Techniken wie Audits und Penetration-Testing (*Test-Hacking*) angeboten. Zudem bietet es Möglichkeiten der Implementierung eines System-Sicherheitsmanagements nach ISO 27001 (ehemals ISO 17799).

5. *Plattform*

Unbekannt.

6. Modell

6.1. *Modellklasse*

Qualitativ, Prozess-Ebene.

6.2. *Modelltyp*

Risikoanalyse und -bewertung nach ISAMM

6.3. *Modellbeschreibung*

0. Phase (Kick-Off-Meeting): ISAMM erklären, Referenz zu ISO 17799:2005 festlegen, Verständnis der Umgebung und Bereiche, Terminplanung für Interviews.

1. Phase (Interviews und Risikoanalyse): Interviews mit Verantwortlichen, Begriffs-definitionen, Worst-Case-Impact-Abschätzung und Festlegung von Angriffsszenarien, Ermittlung der Auftrittshäufigkeit von Bedrohungen und Abschätzung der durch-schnittlichen Verluste.

2. Phase (Sicherheitsmaßnahmen und Empfehlungen): Interview IT-Leitung, Liste von Sicherheitsmaßnahmen aus ISO 17799:2005, Bestimmung der Umsetzungsrate und -kosten. Optionen zum Ausschluss oder der Erzwingung von Maßnahmen. Abgabe von Empfehlungen (Security Policy).

3. Phase (Nachhaltigkeit): Umsetzung von Empfehlungen, Einführung von Bewertungszyklen nach dem PDCA-Modell (*Plan-Do-Check-Act*).

6.4. *Modelliertes System*

IT-Organisationen.

6.5. *Modelleingabe*

Prozessbeschreibung

6.6. *Modellausgabe*

Maßnahmenempfehlungen, Risikobewertung.

6.7. *Schnittstellen*

Keine bekannt.

7. *Anwendungsfälle*

Es wurde eine Sicherheitsanalyse an der Universität Kaiserslautern auf Basis der ISAMM-Methodik durchgeführt. Das Ziel bestand in der allgemeinen Verbesserung der Informationssicherheit durch Behebung von Schwachstellen, Verbesserung der Netzwerk- und Systemzugriffsverwaltung, Vereinheitlichung der Benutzerkontenverwaltung, Verabschiedung von allgemein anerkannten Sicherheitsrichtlinien (Security Policy) und Verbesserung des Sicherheitsbewusstseins der Mitarbeiter. Dabei traten Schwierigkeiten bei

der Umsetzung der Ziele, bedingt durch die ausgeprägte Autonomie universitärer Bereiche, einer inhomogenen Interessenlage und teilweise hohe Technologieanforderungen, bedingt durch Forschung, sowie Organisation der universitären Verwaltung zutage. Das Projekt wurde mit Hilfe einer externen Beratungsfirma durchgeführt. Dabei wurde eine formale Risikoanalyse im Top-Down-Ansatz (ISAMM) auf die universitären Bereiche Rechenzentrum, Fachbereich Mathematik sowie Lehrstuhl für Fertigungstechnik und Betriebsorganisation angewendet.

8. *Annahmen und Einschränkungen*

   Nicht bekannt.

2.4 Tabellarische Zusammenfassung der untersuchten Werkzeuge

In Tabelle 1.2 werden die untersuchten Werkzeuge tabellarisch zusammengefasst und wesentliche Einzelaspekte detailliert.

Name	Lizenztyp	Modellklassen	Modelltyp	Plattformen	Allgemeiner Zweck	Anwendungsfälle
ACARA	Open-Channel-Foundation	Quantitativ – analytisch, IT-Ebene.	Statistische Monte-Carlo-Methoden, Weibull- und Exponentialverteilung, Blockdiagramme, Modelle zum frühzeitigen und zufälligem Ausfall sowie durch Verschleiß.	Windows 98 oder neuer	Analyse von Verfügbarkeit, Lebenszyklus-Kosten und Ressourcenplanung für periodisch reparierte Systeme.	NASA
Aprico Consultans CASIS	Kommerzielle Lizenz,ab € 45.000.	Quantitativ, IT-Ebene.	Correlation-Rules, Datentransformationen.	Unbekannt.	Der Zweck des Werkzeugs liegt in der Sammlung von Logdateien über verschiedene Systeme hinweg, Korrelation dieser Daten und Erzeugung von Sicherheitsalarmen, basierend auf benutzerdefinierten Regeln.	Unbekannt.
ARIES	Unbekannt	Analytisch, IT-Ebene.	Homogene Markov-Modelle, gelöst mittels Lagrange-Sylvester-Interpolation.	APL und C Implementierungen	Zuverlässigkeits- und Lebenszyklus-Analyse für fehlertolerante Systeme.	Zur Unterstützung in der Lehre.
Axis: RA2	Kommerzielle Lizenz (1100 BP), Evaluierungslizenz.	Qualitativ, IT- und Prozessebene	Risikoanalyse und -bewertung nach ISO/IEC 17799 und 27001.	PC/Windows	RA2 ist ein unabhängiges Werkzeug für die Betreibung von Risikomanagement nach den Standards ISO-17799 und ISO-27001. Für jeden Schritt im Prozess bietet das Werkzeug einen entsprechenden Schritt mit anschließender Reportgenerierung der Ergebnisse an.	RA2 wird in einigen mittelständischen Consulting-Unternehmen, unter anderem in Singapur und Japan eingesetzt. Weitere Informationen sind nicht verfügbar.
BQR Care	preiswertere akademische Lizenz; Kommerzielle Lizenz	Analytisch, Simulation, Mechatronik	Ausfallmodus, Auswirkungs- und Gefährdungsanalyse (FMEA/CA), Fehlerbaum-Analyse, Zuverlässigkeits-blockdiagramm	Windows 2000, XP	Das Tool wurde für die Zuverlässigkeitsanalyse von technischen, elektronischen und mechanischen Systemen (Mechatronik) entwickelt.	BQR unterstützt Firmen in den folgenden Industriebereichen: Luft- und Raumfahrt, Automobil, Chemie, Elektronik, Haushalts- und Unterhaltungselektronik, Verteidigung, Militär, Energiewerke, Benzin und Öl, Medizin, öffentliche Verkehrsmittel und Halbleiter
CALLIO Secura 17799	Kommerzielle Lizenz, Akademische Lizenz, Demoversion.	Qualitativ, Prozess- und IT-Ebene.	PDCA(Plan-Do-Check-Act)-Modell, Fragebögen.	PC/Windows mit MS Access	Das Werkzeug dient dazu, Unternehmen bei der Ausrichtung nach den Standards BS 7799 / ISO 17799 unterstützen. Mit dem Werkzeug kann ein Unternehmen ein standardkonformes Information-Security-Management-System (ISMS) aufbauen und betreiben. Das ISMS liefert einen systematischen Ansatz, um sensible Informationen in einem Unternehmen zu verwalten.	Konkrete Anwendungsfälle sind nicht bekannt. Zu den Geschäftskunden von Callio zählen vor allem Firmen aus dem Bereich IT-Consulting und e-Business.
Cara-FaultTree	Kommerzielle Lizenz.; Testversion mit begrenzter Funktionalität.	Analytisch, IT-Ebene.	Fehlerbaumanalyse	Windows 9x, Windows NT, Windows 2000.	Das Tool ist nur auf die Behandlung von Fehlerbäumen spezialisiert. Die Anwendbarkeit des Tools ist dabei begrenzt auf die Anwendbarkeit der Fehlerbaumanalyse.	Vermögensverwaltungs- und Risikomanagementproblemen in Chemie-, Energie-, Verkehrs-, Benzin- und Ölindustrie
CARE III	Unbekannt	Analytisch, IT-Ebene.	Fehlerbäume, nicht-homogene Markov-	VAX-11	CARE III ist ein allgemeines Werkzeug zur Bestimmung	NASA, die Modellierung von

Name	Lizenztyp	Modellklassen	Modelltyp	Plattformen	Allgemeiner Zweck	Anwendungsfälle
			Ketten und Semi-Markov-Ketten.		der Zuverlässigkeit für sehr große, hochverfügbare Systeme.	Systemen im Avionikbereich.
CARMS	Frei verfügbar	Analytisch, IT-Ebene.	Markov-Ketten.	Windows 3.11 und spaeter.	CARMS ist ein integriertes Markov-Modellierungs- und Simulationswerkzeug, um zeitabhängigen, vorhersageorientierten Problemen zu lösen.	Unbekannt.
CASRE/ SMERFS	Open Channel Software	Quantitativ – analytisch, IT-Ebene	Jelinski-Moranda-Modell, Nichthomogenes Poisson-Prozessmodell	Windows, UNIX	Messung und Bewertung der Zuverlässigkeit von Software. CASRE beinhaltet die graphische Darstellung von Ausfalldaten, deren Vorverarbeitung (beispielsweise Filterung) sowie Ausfallprognosen für das entsprechende Softwaremodul.	Software.
COBRA (C&A Systems Security)	Kommerzielle Lizenz, COBRA Suite full/ISO17799 (US$ 1995, 895), Versuchsversion.	Qualitativ, Prozess-Ebene.	Wissensbasierte Risikoanalyse	Unbekannt.	Das Softwarewerkzeug COBRA ermöglicht eine Risikobewertung im Sicherheitsbereich durch das Unternehmen selbst. Es evaluiert die relative Bedeutung aller Bedrohungen und Schwachstellen, und es generiert die geeigneten Lösungen und Empfehlungen dafür.	Unbekannt.
CounterMeasures	Kommerzielle Lizenz: „Enterprise Platform“ ($14500), „Standard Platform“ ($3990) und „Web Survey“ ($2500). Kostenlose Evaluierungslizenz.	Qualitativ, Prozess-Ebene.	Fragebögen, kundenspezifische Prüflisten, Inspektoren. Behandlung: Cost-Benefit-Analysis.	Windows 98/ME/NT/2000/XP, Microsoft Office 2000 oder neuer.	„CounterMeasure“ ist ein Programm zur Betreibung von Risikomanagement unter anderem nach den Vorschriften der US-NIST-800-Serie und der Circular-A-130-Vorschriften des OMB (Office of Management and Budget). Zu den Anwendungsbereiche des Werkzeugs gehören IT-System- und Gebäuderisiken sowie operationales Risiko.	Physische Sicherheit: Verkehrswege, Gebäude, Öl-und-Gas, kritische Infrastruktur. Konkrete Anwendungsfälle sind nicht verfügbar (werden jedoch auf der Webseite angekündigt).
CPNTOOLS	kommerziell, Militär/Regierung und nicht-kommerziell/akademisch.	Analytisch, IT-Ebene.	Farbige Petri-Netze, eine Erweiterung von Petri-Netzen.	Windows 2000, Windows XP, Linux, Fedora Core 2	Bearbeitung, Simulation und Analyse von Farbigen Petri-Netzen (Coloured Petri Nets). Mit Hilfe dieses Modells können insbesondere verteilte Prozesse modelliert werden, die miteinander kommunizieren und sich gegenseitig synchronisieren.	Netzwerkprotokollen, Software, Workflow-Analysen und Geschäftsprozessmodellierung, Steuerungssysteme im Echtzeitbereich, militärische Planung und Flugverkehrskontrolle.
CRAMM	Kommerzielle Lizenz, 30-Tage-Evaluierungsversion.	Qualitativ, Prozess-Ebene	Risikoanalyse und -bewertung nach CRAMM (CCTA Risk Assessment and Management Method).	Nicht bekannt, vermutlich PC/Windows	CRAMM ist ein Softwarewerkzeug zur Automatisierung des CRAMM-Prozesses. Die Methode CRAMM liefert einen stufenbasierten Ansatz, der sowohl technische (IT-Hardware und -software) als auch nicht-technische Aspekte (Mensch, Physis) enthält: Identifizierung eines Aktivpostens und dessen Einschätzung, Bedrohungs- und Schadensanfälligkeitsbewertung sowie eine Auswahl von Gegenmaßnahmen und Empfehlungen.	CRAMM wird benutzt unter anderem von BAE Systems, IBM, General Motors, Royal Air Force, Swiss Bank, T-Mobile. Näher beschriebene Anwendungsfälle sind nicht bekannt.
DyQNtool+	Unbekannt (vermutlich existiert eine	Quantitativ – analytisch, IT-Ebene.	Markovsche Entlohnungsmodelle.	SUN-4-Systeme	DyQNtool+ verwendet das Konzept der dynamischen Warteschlangen-Netzwerke. Als solches basiert das	Unbekannt.

Name	Lizenztyp	Modellklassen	Modelltyp	Plattformen	Allgemeiner Zweck	Anwendungsfälle
	akademische Lizenz).				Werkzeug auf einem Framework, in dem beide Seiten – Aspekte der Zuverlässigkeits als auch der Leistungsfähigkeit sowie ihrer gegenseitigen Abhängigkeit – formal spezifiziert werden können.	
EBIOS	Open-Source	Qualitativ, Prozessebene.	EBIOS(Expression of Needs and Identification of Security Objectives) -Methode.	Windows, Linux, Solaris	EBIOS ist ein Softwarewerkzeug, das die gleichnamige Methode unterstützt. Das Werkzeug erstellt die Risikoanalyse und das Management nach den fünf EBIOS-Phasen, erlaubt die Aufzeichnung aller Ergebnisse und die Erstellung einer Zusammenfassung.	EBIOS wird hauptsächlich im öffentlichen Bereich in Frankreich sowie in Organisationen, bei Consulting-Firmen und Firmen beliebiger Größe in der europäischen Union und Kanada eingesetzt.
Eclipse TPTP	Open Source	Quantitativ - Analytisch, Benchmarking und Test, IT-Ebene.	TPTP Monitoring Tools: Analyse und Vorverarbeitung (Filterung) von Logdateien. Patterns und Matching-Regeln.	Diverse Plattformen (u.a. PC, SPARC) und Betriebssysteme (u.a. Windows, Linux) werden unterstützt	TPTP ist eine standardisierte, generische und erweiterbare Werkzeugplattform, mittels der Softwareentwickler eigene spezialisierte Werkzeuge erstellen können. Zusätzlich liefert TPTP auch Module, mit denen der Anwender bereits grundlegende Performance- und Testaufgaben bewältigen kann.	Software. Unterstützung der Testbeschreibungssprache TTCN-3 für kommunkationsbasierte Anwendungen.
Exhaustif	Kommerzielle Lizenz.	Quantitativ – Benchmarking und Test, IT-Ebene.	SWIFI (Software Implemented Fault Injection).	RTEMS/ERC32, und RTEMS/Intel . Im Zukunft: Windows/Intel und Linux/Intel.	Ausführung von Black-Box- und Grey-Box-Tests basierend auf der SWIFI-Methode (Software Implemented Fault Injection). Das Werkzeug dient dazu, während des Entwicklungsprozesses (u.a während des Integration- und Systemtests) Zuverlässigkeits- und Verfügbarkeitseigenschaften von softwareintensiven Systemen zu verbessern.	Eine Beta-Testvariante soll bei EADS-Astrium verwendet werden.
FAIL-FCI	Unbekannt.	Quantitativ – Benchmarking und Test, IT-Ebene	Softwarebasiertes Verfahren zur Fehlerinjektion.	Linux	FAIL-FCI ist ein Werkzeug zur Zuverlässigkeitsbestimmung von Cluster- bzw. Grid-Systemen. FAIL-FCI erlaubt die Modellierung und Umsetzung von Fehlerszenarien mittels einer abstrakten Fehlerbeschreibungssprache und einem Verfahren zur Fehlerinjektion.	XtremWeb (P2P)
FIGARO/KB3 Workbench	Kommerzielle Lizenz, zeitlich unbegrenzte Testlizenz	Analytisch, Simulation, IT-Ebene	Markov-Ketten, Boolean logic Driven Markov Process (BDMP), Fehlerbäume, Zuverlässigkeitsblockdiagramme, Petri-Netze	Unix/ Xwindow MS Windows	Berechnung der Systemzuverlässigkeit, der Verfügbarkeit und der Leistung mittels vieler unterschiedlicher Modelle. Die Besonderheit dieser Lösung liegt in der Verwendung von Wissensdatenbanken zur Modellierung abstrakter Systeme. Verschiedene Compiler und Übersetzer leiten daraus automatisch die Daten ab, welche für das klassische Zuverlässigkeitsmodell von Bedeutung sind.	Optimierung des Designs oder der Ausnutzung gewerblicher Systeme; Zuverlässigkeitsabschätzung; Sicherheitskontrolle gefährlicher Industrieprozesse (Atomwirtschaft, der Chemie- und Petrochemieindustrie)
Fujitsu Interstage Business Process	Testversion; KommerzielleLizenz	qualitativ, Service-Ebene	Workflow (BPMN, BPEL, XPDL, Wf-XML)	Windows 2000, 2003, XP, Solaris 9, Red Hat	Das Tool verfügt über einen Workflow-basierten Geschäftsprozessmanager, der den gesamten Prozesszyklus	Bank- und Finanzwesen, Energieindustrie, Reisebranche,

Name	Lizenztyp	Modellklassen	Modelltyp	Plattformen	Allgemeiner Zweck	Anwendungsfälle
Manager				Linux, ES 4.0, HP-UX 11i, IBM AIX 5.3	umfasst: Prozessintegrierung, -automatisierung, -modellierung, -verwaltung und -optimierung. Die letzten drei sind dabei für diese Studie interessant, da sie zu einem gewissen Grad Verfügbarkeitsmodellierung / Bewertungsmöglichkeiten bieten.	Unternehmens-software und Verkehrsregelung
GRAMP/ GRAMS	Unbekannt.	Simulation, IT-Ebene.	Zeitkontinuierliche Markov-Modelle (GRAMP), gelöst mittels der diskreten Monte-Carlo-Simulation (GRAMS).	FORTRAN-77-Implementierung für VMS.	GRAMP: Fehleraufdeckung, vorbeugende Wartung, Anschaffungskosten, Betriebskosten, Support-Kosten, Empfindlichkeitsanalyse. GRAMS: Vorhersage der Zuverlässigkeit, Wartbarkeit und Kosten des Lebenszyklus eines Systems, welches von GRAMP modelliert wurde.	Fehlertoleranten elektronischen Controllern (Full-Authority Fault-Tolerant Electronic Engine Control (FAFTEEC) Program)
GSTOOL	Kostenfrei fuer unmittelbare Bundes-, Landes- und Kommunalverwaltung der Bundesrepublik Deutschland	Qualitativ, Prozess- und IT-Ebene.	Risikoanalyse nach dem BSI-Grundschutz (BSI-Standard 100-3).	PC/Windows	Mit dem Werkzeug stellt das BSI eine Software bereit, die den Anwendern bei Erstellung, Verwaltung und Fortschreibung von IT-Sicherheitskonzepten entsprechend dem IT-Grundschutz effizient unterstützt.	Das Werkzeug wird in der Verwaltung des Bundes, der Länder und Kommunen sowie im Unternehmensbereich verwendet. Detaillierte Beschreibungen sind nicht verfügbar.
HARP	Unbekannt.	Analytisch, IT-Ebene.	FORM (Fault Occurence and Repair Model), FEHM (Fault/Error Handling Model), Markov-Ketten	MS/PC-DOS, Microsoft Windows NT, OS/2, DEC VMS und Ultrix, Berkeley UNIX 4.3 und AT&T UNIX 6.2.	Vorhersage von Zuverlässigkeit und Verfügbarkeit. Als Eingabe dient ein Fehlerbaum oder graphische Eingabe durch den Benutzer.	Unbekannt.
IBM High Availability Services	Nicht verfügbar.	qualitativ, Service-Ebene	Fragebogen	Nicht verfügbar.	IBM High Availability Services helfen bei der Vermeidung teurer Ausfälle und Wiederherstellungen. Sie unterstützen Bestimmung, Planung, Entwurf, Konstruktion, Implementierung und Verwaltung einer Infrastruktur, welche dauerhafte Geschäftsverfügbarkeit und Einhaltung von Service-Level-Vorgaben ermöglicht.	Nicht verfügbar.
IBM Tivoli Availability Process Manager	Kommerzielle Lizenz; Testversion	qualitativ, IT- und Service-Ebene	Component Failure Impact Analysis (CFIA) und Verfügbarkeitsverwaltung (Verwaltung von Vorfällen), wie durch ITIL spezifiziert.	IBM AIX 5.2, 5.3, RedHat Linux, Microsoft Windows	Der Tivoli Availability Process Manager erlaubt die Verfügbarkeitsbestimmung und die Prioritätszuweisung bei Vorfällen entsprechend ihrer Auswirkungen sowohl auf die IT-Infrastruktur als auch auf kritische Geschäftsprozesse. Auf diese Weise erzeugt das Tool eine Verbindung zwischen der IT- und der Geschäftsebene. Es bietet auf ITIL basierende Verfügbarkeitsverwaltung (Verwaltung von Vorfällen) und bestimmt die Auswirkung eines Ausfalls auf das Geschäft.	Automobilindustrie, Bankwesen, Computer-Dienstleister, Energiewirtschaft, Behörden, Gesundheitswesen, Medien.

Name	Lizenztyp	Modellklassen	Modelltyp	Plattformen	Allgemeiner Zweck	Anwendungsfälle
ISAMM	Unbekannt, Werkzeug ist nicht verfügbar.	Qualitativ, Prozess-Ebene	Risikoanalyse und -bewertung nach ISAMM (Information Security and Monitoring Method).	Unbekannt.	ISAMM ist ein Werkzeug zur Betreibung von Risikomanagement. Es berechnet einen idealen Sicherheits-Sanierungsplan, der alle bedeutsamen Aktionen zur Erhöhung der Sicherheit nach der Wirtschaftlichkeit – dem ROSI-Wert (Return On Security Investment) – berechnet.	Es wurde eine Sicherheitsanalyse an der Universität Kaiserslautern durchgeführt.
IsoGraph AttackTree+	Testversion; KommerzielleLizenz	analytisch, Simulation, Service-Ebene	Attack-Tree	Microsoft Windows	in Attack-Tree ermöglicht die präzise Modellierung von Gefährdungen der Systemsicherheiten in einem graphischen Format.	Internetanwendungen, Bankwesen, Netzwerke.
IsoGraph AvSim+	Testversion; KommerzielleLizenz	Simulation, IT-Ebene	Fehlerbaum, Netzwerk- (Zuverlässigkeitsblock-) Diagramm	Microsoft Windows	Verfügbarkeits- und Zuverlässigkeitssimulation komplexer, abhängiger Systeme.	Luft- und Raumfahrt, Verteidigung, Automobil-, Eisenbahn-, Chemie-, Benzin- und Ölindustrie, Medizin
IsoGraph FaultTree+	Testversion; KommerzielleLizenz	analytisch, IT-Ebene	Fehler- und Ereignisbäume, Markov-Modelle	Microsoft Windows	Das Tool bietet eine Zuverlässigkeitsanalyse, so dass Fehler- und Ereignisbaumanalysen durchgeführt werden können. Maßgeschneiderte Markov-Modelle können mit Ereignissen im Fehler- oder Ereignisbaumdiagramm in Zusammenhang stehen. Es ist allerdings auch möglich voneinander unabhängige Analysen zu verwirklichen.	Luft- und Raumfahrt, Verteidigung, Automobil-, Eisenbahn-, Chemie-, Benzin- und Ölindustrie, Medizin
IsoGraph NAP	Testversion; KommerzielleLizenz	analytisch, IT-Ebene	erweitertes Zuverlässigkeitsblockdiagramm	Microsoft Windows	NAP ermöglicht die Vorhersage von Verfügbarkeit und Zuverlässigkeit von Kommunikationsnetzwerken.	Kommunikationsnetzwerke
IsoGraph Reliability Workbench	Testversion; KommerzielleLizenz	analytisch, IT-Ebene	verschiedene Vorhersagemodelle, hierarchische Blockdiagramme	Microsoft Windows	Reliability Workbench ist eine integrierte Umgebung, mit der Zuverlässigkeits- und Wartungsvorhersagen, FMECA (Failure Mode, Effect and Criticality Analysis), Zuverlässigkeitsblockdiagramm-, Fehlerbaum-, Ereignisbaum- und Markov-Analysen sowie Zuverlässigkeitszuordnungen gemacht werden können.	Luft- und Raumfahrt, Verteidigung, Automobil-, Eisenbahn-, Chemie-, Benzin- und Ölindustrie, Medizin
ItSMF (Self-Assessment)	Frei	Qualitativ, Prozess-Ebene	Fragebogen	Online/MS Excel	Der Zweck des Fragebogens ist es, der Organisation (und ihrem Management) ein Idee zu geben, wie gut der derzeitige Stand im Vergleich zum ITIL-Best-Practice-Ansatz ist.	IT-Consultant-Sektor; Regierungsorganisationen
MARK 1	Unbekannt.	Analytisch, IT-Ebene.	Zustandsdiskrete, zeitkontinuierliche Markov-Modelle.	PL/1	Auswertung der Zuverlässigkeit von komplexen Systemen, deren Eigenschaften mittels Markov-Ketten modelliert werden können. Das Werkzeug berechnet Zustandswahrscheinlichkeiten als Funktionen über die Zeit, MTBF sowie die Wahrscheinlichkeit der durchschnittlichen Verbleibensdauer in einem Zustand (Average State Occupancy).	Fehlertolerante Multiprozessoren (FTMP) , entwickelt für die NASA, Flugzeugsysteme und Betriebsicherheitssysteme in Kernkraftwerken.
Mathworks Stateflow	Kommerzielle Lizenz,	Quantitativ – analytisch, IT-	State-Charts, endliche Zustandsautomaten	Windows 2000, XP,	Stateflow ist ein Simulationswerkzeug für ereignisgesteuerte	Fehlertoleranten Kraftstoff-

Name	Lizenztyp	Modellklassen	Modelltyp	Plattformen	Allgemeiner Zweck	Anwendungsfälle
	kostenpflichtige akademische Lizenz, kostenlose 15-Tage-Demoversion	Ebene	(Finite State Machines), temporale Logik.	Vista, Linux, Solars 8.x und spaeter, MacOS auf Intel.	Systeme. Stateflow erweitert herkömmliche State-Charts um folgende Konzepte: Kontrollfluss, Wahrheitstabellen, temporale Operatoren, ereignisgerichtetes Senden (Broadcasting).	Füllsystems, zahlreiche Anwendungsfälle in der Industrie.
Mercury BTO Enteprise Solutions	Kommerziell; Versuchsversion	Qualitativ, IT- und Prozess-Ebene	ITIL-Service-Managements; COBIT Qualitätsstandard	Windows, Linux/Unix	Mercury BTO (Business Technology Optimization) Enterprise ist eine Werkzeugsuite, die den Anwender bei der Implementierung des ITIL-Service-Managements unterstützt. Dazu werden Techniken wie Dashboards, CMDB (Configuration Management DataBase) und eine Workflow-Engine zur Verfügung gestellt. Mercury Project and Portfolio Management bietet auch die Unterstützung von COBIT und weiteren Qualitätsstandards an.	Nicht verfügbar
METASAN	Unbekannt	Analytisch, IT-Ebene.	Stochastic Activity Networks (SAN)	UNIX	Evaluierung der Leistung und Zuverlässigkeit von Systemen durch Analyse und Simulation.	Unbekannt.
METFAC	Lizenzfreie 30-Tage-Evaluierungsversion mit voller Funktionalität, lizenzfreie voll-funktionsfähige Version für Grundlagenforschung, kommerzielle Lizenz.	Analytisch, IT-Ebene.	Finite zeitkontinuierliche Markov-Ketten-Modelle mit bewerteten zustandsabhängigen Raten.	Linux-Kernel 2.4.4 und darüber; Solaris 2.x; bei Bedarf jede andere Unix-Variante mit C-Shell und einem ANSI/ISO-Compiler (Standard 89/90).	Analyse der Performance, Zuverlässigkeit und Performabilität komplexer Systeme durch bewertete, zeitkontinuierliche Markov-Ketten-Modelle.	Zuverlässigkeitsmodelle: 5-Level-RAID-Speichersubsystem; Speichersystems; Performancemodelle: Multiprozessor-Systems; GRID-Cluster-Computersystems; Speichernetzwerkes (Storage Area Network); Kommunikationsnetzwerkes.
Möbius	Akademische und kommerzielle Lizenz, 30-Tage Evaluierungs-version	Quantitativ – analytisch, IT-Ebene.	Stochastic Activity Networks (SAN), Warteschlangentheorie, stochastische Petri-Netze, stochastische Prozessalgebren (SPA), stochastische Automatennetzwerke, Fehlerbäume, kombinatorische Blockdiagramme.	Windows2000, XP, Fedora Core 3 und später.	Modellierung des Verhaltens komplexer Systeme. Zuverlässigkeit-, Verfügbarkeit- und Leistungsfähigkeitanalyse von Computer- und Netzwerksystemen.	Systeme der Informationstechnologie und Netzwerke; Draht- und drahtlose Telekommunikationssoftware- und Hardware-Systeme; Systeme der Luft- und Raumfahrt; Biologische Systeme
NFTAPE	Akademische Lizenz.	Quantitativ – Benchmarking und Test, IT-Ebene.	Software-basierte Fehlerinjektion.	Solaris, Linux	NFTAPE (Networked Fault Tolerance and Performance Evaluator) ist eine softwarebasierte Umgebung zur automatischen Bestimmung der Zuverlässigkeit eines Netzwerkes auf Basis der Fehlerinjektionsmethode.	Motorola IDEN MicroLite (Base-Station-Controller), DHCP (Dynamic Host Configuration Protocol), SIFT-Umgebung für die REE-Testeinbettung, Evaluierung der Schadensanfälligkeit von SSH- und FTP-Diensten.
NUMAS	Nicht verfügbar.	analytisch, IT-Ebene	Warteschlangennetze, Markov-Ketten	Sun	NUMAS ist ein Leistungsanalyse-Tool, bei dem jeder	Nicht verfügbar.

Name	Lizenztyp	Modellklassen	Modelltyp	Plattformen	Allgemeiner Zweck	Anwendungsfälle
					Knotenpunkt mit Fehlertoleranzcharakteristiken erweitert werden kann.	
OCTAVE (Operationally Critical Threats, Assets, and Vulnerability Evaluation) Automated Tool	Kommerzielle Lizenz ($1500). Demonstrations -und Versuchsversion nicht verfügbar.	Qualitativ, IT- und Prozessebene.	Risikoanalyse	Unbekannt.	Das Werkzeug wurde von Advanced Technology Institute implementiert, um dem Benutzer die Anwendung des Octave-Ansatzes zu erleichtern. Das Werkzeug assistiert dem Benutzer während der Datensammlungsphase, der Datenorganisation und dem Erstellen eines Abschlussberichtes.	Gesundheitswesen (OCTAVE - HIPAA, OCTAVE - JCAHO)
OpenSESAME	Akademische Lizenz, Kommerzielle Lizenz auf Nachfrage.	Quantitativ – analytisch, IT-Ebene.	Zuverlässigkeits-Blockdiagramme, Fehlerabhängigkeitsdiagramme, Stochastische Petri-Netze.	Linux/UNIX	OpenSESAME kombiniert die benutzerfreundliche Modellierung von kombinatorischen Techniken (RBD) mit der Ausdrucksstärke zustandsbasierter Modellierungstechniken (Petri-Netze). Die gegenseitige Abhängigkeit von Komponenten kann auch spezifiziert werden.	Webserver; Telekommunikationsschaltstellen
PENELOPE	Unbekannt.	Quantitativ – analytisch, IT-Ebene	Extended Markov Reward Model (EMRM), Controlled Stochastic Petri Nets (COSTPN)	SUN-4	Performanceanalyse und Optimierung.	Beispielstudie eines Bereitstellungsmodells für den Gefahrenfall (Emergency Supply Model).
PENPET	Unbekannt.	Quantitativ – analytisch, qualitativ, IT-Ebene.	GSPN (Generalized Stochastic Petri Nets), Markovsche Entlohnungsmodelle.	C-Implementierung unter UNIX	Analyse der Leistungsfähigkeit und Zuverlässigkeit von fehlertoleranten Systemen.	Multiprozessor-Systemen.
PILAR/EAR	PILAR darf nur innerhalb der spanischen Administration verwendet werden, von EAR gibt es eine Public-Domain-Lizenz im Read-Only-Modus und eine Benutzerlizenz für das Programm im Write-Modus.	Qualitativ, Quantitativ-analytisch., Prozess-Ebene.	MAGERIT (Methodology for Information Systems Risk Analysis and Management)-Methode mittels Tabellen sowie Attack-Trees, Prozessbeschreibungen, Audits. Quantitativ-analytische Ansätze: Datenflussdiagramme, Process-Charts, Boolesche Funktionen.	Windows, Linux	PILAR ist ein Werkzeug, das das MAGERIT-Verfahren des spanischen Verteidigungsministeriums implementiert und erweitert. Die Funktionalität umfasst die Bereiche quantitative und qualitative Risikoanalyse sowie Risikomanagement; Business-Impact-Analysis-and-Continuity-of-Operations.	PILAR: innerhalb der spanischen Verwaltung
PROTEUS	Kommerzielle Lizenz (PROTEUS Solo 599 BP/Jahr, PROTEUS Professional 6000 BP/Jahr), WebEx-Demo.	Qualitativ, IT- und Prozessebene	Risikoanalyse und -bewertung.	Client: Microsoft Windows NT/2000/XP Server: Microsoft Windows Server 2003, Linux	PROTEUS ist ein Webserver-basiertes Werkzeug zur Unterstützung des Informationssicherheits- und Risikomanagements und der Überprüfung der Einhaltung von Standards (ISO/IEC 17799 und BS ISO/IEC 27001, BS 25999, COBIT, PCI DSS, etc.).	Unbekannt.
QUAKE	Nicht verfügbar.	Quantitativ – Benchmarking	Fehleinjektion, Zeitreihenanalyse (Time-	Nicht verfügbar.	Bestimmung der Zuverlässigkeit von Grids und Web-	Grids und Webservices. Details sind

Name	Lizenztyp	Modellklassen	Modelltyp	Plattformen	Allgemeiner Zweck	Anwendungsfälle
		und Test, IT-Ebene.	series Analysis), Benchmark.		Services.	nicht verfügbar.
Relex Reliability Studio, PRISM	Kommerzielle Lizenz, Evaluierungsversion	Quantitativ – analytisch, qualitativ, IT- und Prozessebene.	Markov-Modelle, Zuverlässigkeits-Blockdiagramme (RBD) mit Monte-Carlo-Simulation, FMEA/FMECA, FTA/ETA, FRACAS, Human-Factor-Risk-Analysis.	Windows	Reliability Studio ist eine Werkzeugzusammenstellung der Firma Relex, um Aussagen über die Zuverlässigkeit, Performance und Wartbarkeit von Computersystemen zu treffen. Es werden Methoden und Vorgehensmodelle zur Evaluierung der Kritikalität von Fehlem und sowie Modelle zur Analyse der Zuverlässigkeit und Verfügbarkeit angeboten.	Luft- und Raumfahrt, Automobil, Öl und Gas, Medizintechnik, Eisenbahn und Telekommunikation.
Reliability Center: Proact, LEAP	Kommerzielle Lizenz.	Quantitativ – analytisch, qualitativ, IT-Ebene.	Root-Cause-Analysis-Methode. FMEA und Opportunitätsanalyse.	PC/Windows	PROACT ist eine gleichnamige Softwarelösung für das Proact-Prozessmodell, einer Variante der Root-Cause-Analysis-Methode. LEAP ist ein Werkzeug zur Umsetzung der FMEA- und Opportunitäts-Methode.	Luft- und Raumfahrt (NASA), Eisenbahn (Amtrak), Lebensmittelindustrie (Bacardi), Multikonzerne (General Electric), Automobil (General Motors), Elektrizitätswerke (Virginia Powers), Öl und Gas (Shell).
Reliasoft	Kommerzielle- und Evaluierungslizenz	Quantitativ – analytisch, qualitativ, IT-Ebene.	Life Data Analysis, RCM, FMEA/FMECA, RCM, RBD, FRACAS. Stochastische Ereignissimulation.	PC/Windows	Die Werkzeuge dienen dazu, den Bereich der Verfügbarkeitsanalyse von Software (bis auf MPC 3 und Lambda Predict) abzudecken.	System- und Produkterstellung, Umwelt-Ressourcenplanung und -steuerung, Wirtschaftsmodellierung.
RELIASS	Kommerzielle Lizenz, Evaluierungsversion	Quantitativ – analytisch, IT-Ebene.	FMECA, RCM, Korrektive und vorbeugende Wartung, Thermal-Analysis, Sensitivitätsanalyse (ASENT), RBD, Monte-Carlo-Simulation, Markov-Ketten, Fehlerbaum- und Ereignisablaufanalyse, Weibull-Ausfallverteilungen, Zuverlässigkeitsvorhersage, Weak-link-Analysis und Phased-Simulation	Windows 98 und spaeter.	Reliass hat sich auf den Verkauf von e-Commerce-Software für Web-basierte Anwendungen spezialisiert. Zusätzlich werden auch Skripte für Computersicherheit angeboten.	Lockheed Martin für das JSF-Projekt (F-35 Kampfflugzeug)
RemedySuite	Kommerzielle Lizenz	Qualitativ, Prozess-Ebene	Remedy Software für IT-Service-Management (ITSM)	Windows-Famile sowie Linux- und Unix-Betriebssysteme	Das Werkzeug dient dazu, Verbesserungen in der Qualität des Projektmanagements über die gesamte Projektlebenszeit zu erreichen. Das Werkzeug soll ITIL-Prozesse in großen Unternehmen verwalten und automatisieren.	BMC unterstützt eine lange Liste an Firmen, die die Remedy-Suite oder Teile davon benutzen. Einige der bekannten Firmen davon sind: TeliaSonera, Infenion, Vodafone Egypt, Agfa, Dell.
RiskWatch	Kommerzielle Lizenz, . Online-Repräsentation.	Qualitativ, IT- und Prozessebene.	Risikoanalyse und -bewertung.	PC/Windows XP/Office Professional	Das Werkzeug dient zur automatischen Risikoanalyse und Bewertung der Schadensanfälligkeit von	Unbekannt.

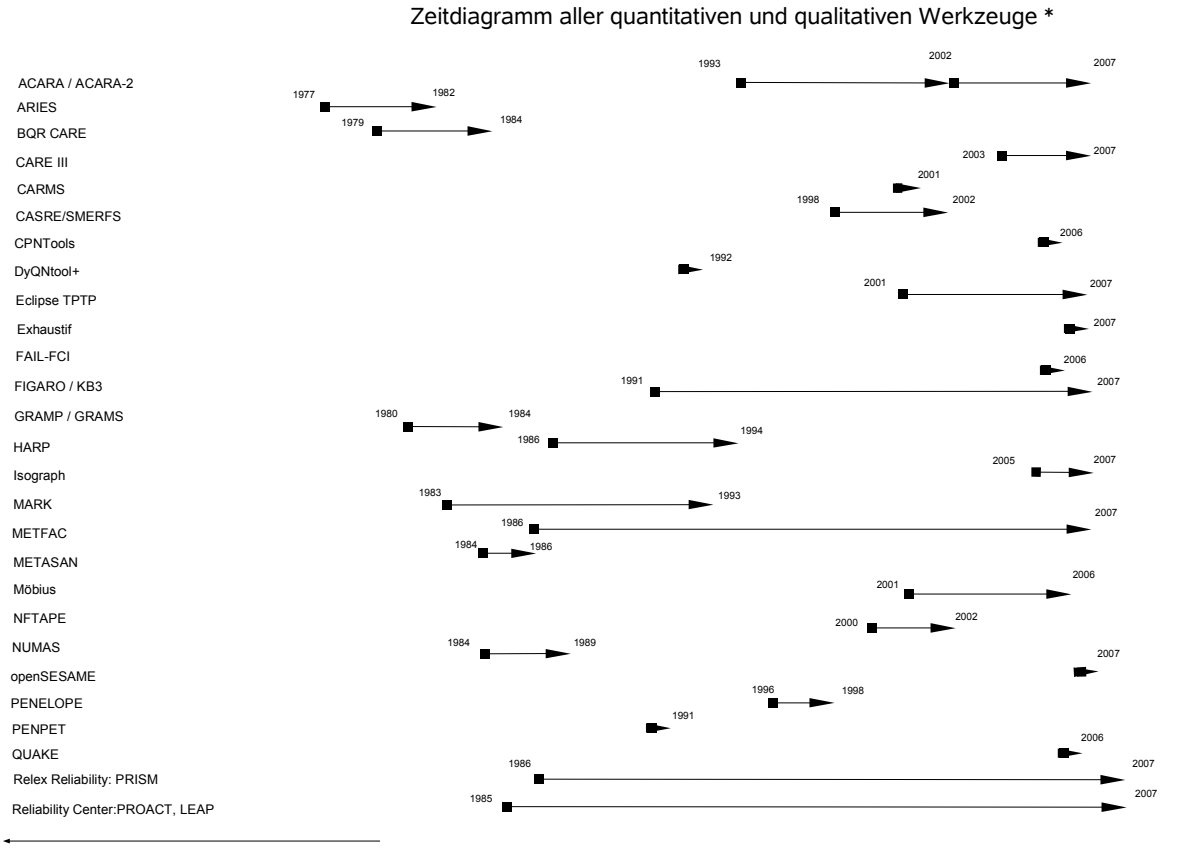
Name	Lizenztyp	Modellklassen	Modelltyp	Plattformen	Allgemeiner Zweck	Anwendungsfälle
					Informationssystemen. Die mitgelieferte Wissensbasis ist kundenspezifisch anpassbar.	
Sabaton	Kommerzielle Lizenz; Testversion mit begrenzter Funktionalität	Analytisch, IT-Ebene	FMEA/CA	Windows 9x, Windows NT, Windows 2000	Sabaton unterstützt FMEA (Failure Mode and Effects Analysis) und FMECA (Failure Mode, Effects and Criticality Analysis), die üblicherweise in der Produkt- und Systementwicklung zur Aufdeckung von möglichen Ausfällen und Ausfallarten sowie zur Abschätzung der Ausfallfolgen verwendet werden.	Verteidigungssektor und zivilen Umfeld;
SAVE	Nicht verfügbar	Analytisch und simulativ, IT-Ebene.	Homogene Markov-Ketten.	FORTRAN 77	Lösen von probabilistischen Modellen der Systemverfügbarkeit und -zuverlässigkeit von einsatzorientierten und kontinuierlich betriebsbereiten Systemen.	Computer in der Luft- und Raumfahrt und Telefonschaltsysteme, Mehrzweckcomputer, Transaktionssysteme.
SHARPE 2000/2002	Akademische Lizenz, Kommerzielle Lizenz	Analytisch und simulativ, IT-Ebene.	Markov-Ketten (irreduzibel, azyklisch, phasentypisch), Semi-Markov-Kettten, RBD, Fehlerbäume, Zuverlässigkeitsgraphen, Single/Multi-Chain-Product-Form-Warteschlangennetze, generalisierte, stochastische Petri-Netze, seriell-parallele Graphen.	Windows, Linux, Solaris, JVM	SHARPE 2000/2002 ist eine Werkzeugzusammenstellung; enthalten darin sind eine Spezifikationssprache und Lösungsmethoden der meisten aller gewöhnlich benutzten Modelltypen zur Modellierung von Performance, Zuverlässigkeit und Performabilität.	Auf der Projektwebseite wird behauptet, dass Softwarepakete an 280 Orten installiert sind, jedoch wurden bis jetzt keine publizierten Anwendungsfälle dazu gefunden.
SoftRel LLC: Frestimate	Standard und Manager Edition; Metrics Package	Analytisch, Software	Angepasstes Modell basierend auf Korrelation	Windows 2000, Windows XP	Das Tool wurde zur Einschätzung der Softwarezuverlässigkeit hinsichtlich der zu erwartenden Anzahl von Fehlem pro 1000 Codezeilen entwickelt.	Halbleiter-, Verteidigungs-, Luft- und Raumfahrtindustrie
Software AG CentraSite	Kommerzielle Lizenz; Testversion	qualitativ, Service-Ebene	Metadaten-Speicher, die Policy unterstützt, Änderungssteuerung, Wartung, Automatisierung, Abhängigkeitsanalyse und Auswertung auf Geschäftsprozess-(Service-) Ebene.	Plattformunabhängig (Java- und Browser-basierte graphische Benutzoberfläche)	CentraSite ist eine Plattform für eine Serviceorientierte Architektur (SOA) Governance.	Bekannte Nutzer sind beispielsweise Belgian National Railway Company, Commerzbank, DaimlerChrysler, Nissan Motors, Scandinavian Airlines, Vodafone, Volkswagen und ZDF.
SPNP	Akademische sowie kommerzielle Lizenz auf Nachfrage.	Quantitativ – analytisch, IT-Ebene.	SRN (Stochastic Reward Net), FSPN (Fluid Stochastic Petri Nets).	MS-DOS, Solaris, Linux	Das SPNP (Stochastic Petri Net Package) ist ein vielfach einsetzbares Modellierungswerkzeug zur Analyse der Performance, Zuverlässigkeit und Performabilität von komplexen Systemen.	Verschiedene exemplarische Anwendungsbeispiele (z.b. Datenbanksysteme, ATM-Netzwerke)
SURE	Unbekannt.	Analytisch, IT-Ebene.	Semi-Markov-Ketten	Solaris, Linux, Windows 98	Zuverlässigkeitsanalyse von fehlertoleranten Architekturen. Berechnet untere und obere Grenzen für die Wahrscheinlichkeit, dass das System läuft oder ausgefallen ist.	Unbekannt.
SURF-2	Kommerzielle Lizenz. Preisgünstige	Analytisch, IT-Ebene.	Markov-Ketten, GSPN (Generalized Stochastic Petri Nets).	SUN OS 4.1.x oder SOLARIS 2.x	Verlässlichkeit von Hardware und Software. Die Modellierung erfolgt entweder mit Markov-Ketten oder	Unbekannt.

Name	Lizenztyp	Modellklassen	Modelltyp	Plattformen	Allgemeiner Zweck	Anwendungsfälle
	akademische Lizenz				GSPN (Generalized Stochastic Petri Nets).	
TANGRAM	Unbekannt.	Abhängig von Einsatzgebiet.	Abhängig von Einsatzgebiet.	SUN-3	TANGRAM liefert eine schichtartige Werkzeugumgebung, die auf spezielle Anwendungsbereiche zugeschnitten werden kann, vorausgesetzt die geeigneten Evaluierungstechniken sind verfügbar.	Unbekannt.

Tabelle 1.2: Tabellarische Übersicht aller durchgesehenen Werkzeuge.

2.5 Darstellung der Werkzeuge in chronologischer Sicht

In den Abbildungen 3 und 4 wird eine chronologische Übersicht der Werkzeuge zur Verfügbarkeitsermittlung gegeben.



* Zugrunde gelegt wurden alternativ erst- und letztmaliges Software-Release, Publikationen sowie Unternehmensgründung und Aktualisierungen der Webseite.

Abbildung 3: Chronologische Übersicht von Werkzeugen zur Verfügbarkeitsermittlung (Teil 1).

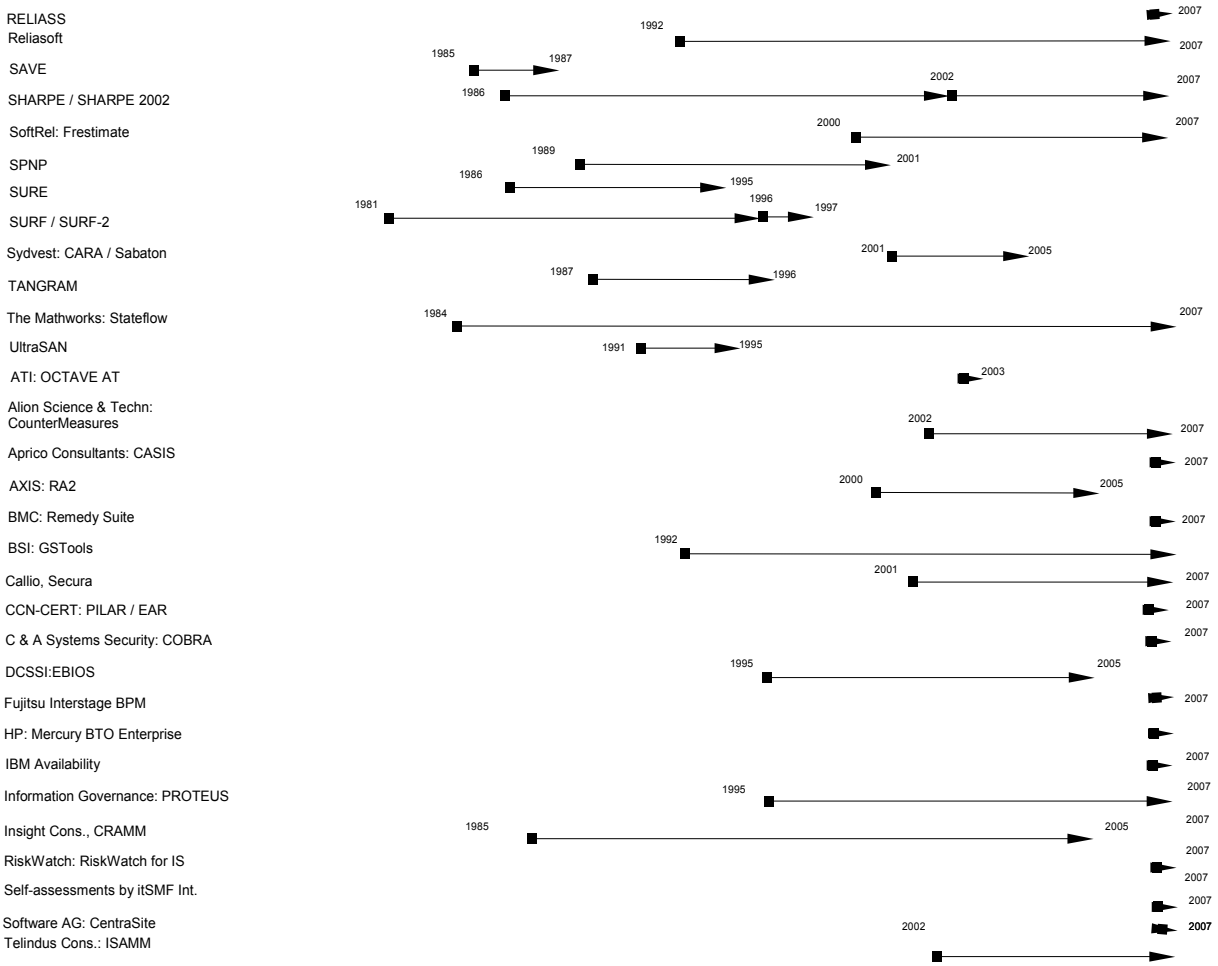


Abbildung 4: Chronologische Übersicht von Werkzeugen zur Verfügbarkeitsermittlung (Teil 2).

2.6 Diskussion und Auswertung der Ergebnisse

Die untersuchten Werkzeuge werden innerhalb der folgenden Gruppen miteinander verglichen:

- Quantitative Werkzeuge (analytisch, Simulation, Benchmarking),
- Qualitative Werkzeuge (Risikoeinschätzung, Prozessverwaltung) und
- Hybride Werkzeuge.

Weiterhin werden die Kompatibilität und Nutzbarkeit der Werkzeuge behandelt. Zum Abschluss erfolgt eine Zusammenfassung der Untersuchungsergebnisse.

Quantitative Werkzeuge

Quantitative Werkzeuge dienen der Durchführung einer quantifizierbaren Verfügbarkeitsbestimmung durch Systemmodellierung bzw. durch die Erstellung eines Modells unter Verwendung analytischer Methoden oder Simulationen. Eine gesonderte Gruppe sind Benchmarking-Werkzeuge.

2.6.1.1 Analytische und Simulations-Werkzeuge

Der Entwurf und die Implementierung von analytischen und Simulations-Werkzeugen ist einfach und geradlinig hinsichtlich Eingabe und Ausgabe. Der überwiegende Teil dieser Werkzeuge basiert auf Konzepten akademischer Werkzeuge wie z.B. ARIES, GRAMP/GRAMS, SURF, METASAN und HARP. Zu den gemeinsamen Eigenschaften gehört eine gute Unterstützung für statische, offline und formale Modellierung fehlertoleranter Systeme. Hierzu werden eine Vielzahl von Methoden und Modellen unterstützt, wie beispielsweise Markov-Ketten, Petri-Netze, Zuverlässigkeits-blockdiagramme und Fehlerbäume. Auch die Ausgabe der Werkzeuge unterscheidet sich nur geringfügig. Sie umfasst verschiedene Steady-State-Verfügbarkeits-, Zuverlässigkeits- und Durchführbarkeitsmetriken, die in den betreffenden Werkzeugen genauer beschrieben werden. Die relevanten Werkzeuge sind: Isograph (FaultTree+, AvSim+, Reliability Workbench, NAP und AttackTree+), METFAC, Möbius, OpenSESAME, Reliass, SHARPE 2002, SPNP, FIGARO, BQR und Mathworks Stateflow.

Isograph und SHARPE 2002 sind umfangreiche Werkzeuge, die eine große Anzahl von Modellen innerhalb eines einzelnen Frameworks integrieren. Allerdings können diese nicht ausgetauscht werden, d.h., Modelltransformationen müssen manuell durchgeführt werden.

Isograph bietet eine rudimentäre Ausfallvorhersage, welche statisch ist und auf vordefinierten Daten

und Protokollen von Ausfallquoten beruht. Diese sind überwiegend für elektrische Geräte spezifiziert. Des Weiteren unterstützt Isograph explizit die Modellierung von Kommunikationsnetzwerken und Angriffen, was mit SHARPE 2002 nur implizit möglich ist.

SHARPE 2002 ist eng an SPNP gekoppelt, welches Petri-Netz-basierte Werkzeuge anbietet sowie eine Vielzahl von Reward-Modellen inkludiert. Zusammen bilden diese beiden Werkzeuge eine Produkteinheit, die mit der Komplexität und der Mächtigkeit von Isograph vergleichbar ist. Beide bieten analytische Methoden und Simulationsverfahren (Monte-Carlo) zur Modelllösung an. Während SHARPE 2002 und SPNP akademische Entwicklungen sind, ist Isograph ein kommerzielles Werkzeug.

METFAC ist ein akademisches Produkt, das auf Markov-Modelle beschränkt ist, und daher nicht direkt mit Isograph oder der SHARPE/SPNP Kombination verglichen werden kann.

Möbius ist ebenfalls ein akademisches Werkzeug. Es verbindet verschiedene Modellierungsarten, wie beispielsweise Warteschlangen, Petri-Netze und Fehlerbäume. Der Unterschied zu Isograph und SHARPE besteht darin, dass mit Möbius verschiedene Problemaspekte mit unterschiedlichen Modellierungstechniken integriert und kombiniert werden können. Zudem löst Möbius die Modelle mit Hilfe analytischer Techniken und Simulationstechniken. Die Fähigkeit unterschiedliche Modellierungs-techniken zu kombinieren ist ein wichtiger Aspekt für die vorliegende Studie. Es ist geplant, die Funktionsweisen von Möbius zu untersuchen, um seine Fähigkeit zur Modellintegration zu prüfen.

Die OpenSESAME-Werkzeuge basieren auf limitierten Modelltransformationen. So wird z.B. ein Zuverlässigkeitsblockdiagramm als Eingabe akzeptiert und dann intern in ein stochastisches Petri-Netz umgewandelt. Obwohl dieser Prozess automatisiert ist, ermöglicht OpenSESAME es nicht verschiedene Einsatzgebiete durch unterschiedliche Modellierungsverfahren abzubilden.

Reliass ist ein integriertes Produkt, welches auch eine Zuverlässigkeitsbestimmung ermöglicht. Das ist jedoch nicht die primäre Aufgabe des Werkzeugs. Reliass besteht aus einer Vielzahl von Modulen, von denen nur einige die Modellierung von fehlertoleranten Systemen ermöglichen. Hierzu werden Zuverlässigkeitsblockdiagramme und Markov-Modelle eingesetzt. Die Ergebnisse dieser Analyse gehen z.B. in die Verwaltung und die Durchführung von Gefährdungsanalysen ein.

Dieser Ansatz ist vergleichbar mit qualitativen Werkzeugen im Bereich der Prozessverwaltung und -optimierung. Zusätzlich konzentriert sich Reliass auf die IT-Infrastruktur-Ebene und nicht auf die Prozess- oder Service-Ebene.

Das BQR-CARE Werkzeug ist Teil eines größeren Analyse-Baukastens und spezialisiert sich auf Failure Mode, Effects and Criticality Analysis (FMEA/CA) sowie Analysen mittels Fehlerbäumen und Zuverlässigkeitsblockdiagrammen. Die Kritikalitätsanalyse ist auf die IT-Ebene und auf vordefinierte Bibliotheken für Anwendungen in der Elektronik und Mechatronik beschränkt.

Mathworks Stateflow ist eine Mehrzweck-Simulationsumgebung, die zur Modellierung und Simulation fehlertoleranter Systeme verwendet werden kann. Dieses Werkzeug bietet keine Verfügbarkeits-spezifischen Erweiterungen, allerdings können Verfügbarkeits-relevante Metriken unter Verwendung von Statecharts, endlichen Automaten und temporaler Logik simuliert werden. Der Stateflow-Simulator ist leistungsfähig, d.h. er erlaubt die Abbildung einer großen Anzahl von Zuständen, die Abbildung komplexer Systeme, eine schnelle Auswertung und eine gute Lösungsqualität.

Bei FIGARO wird das untersuchte System mittels einer abstrakten, objektorientierten Sprache modelliert. Diese wird dann automatisiert in ein Modell überführt, z.B. in ein Petri-Netz, ein Markov-Modell, ein Zuverlässigkeitsblockdiagramm oder einen Fehlerbaum. Der Vorgang wird über eine Regeldatenbank gesteuert und ist gegenüber dem Benutzer vollständig transparent. FIGARO ist mit der KB3-Workbench gekoppelt. Diese übernimmt die von FIGARO generierten Modelle als Eingabewert und verarbeitet diese analytisch oder unter Verwendung von Simulationsmethoden weiter. Dieser Ansatz benötigt kein Wissen über die Modellierungssprache und die zugrunde liegenden Formalismen.

Analytische- und Simulationswerkzeuge bieten hilfreiche Mechanismen zur Modellierung Verfügbarkeits-relevanter Eigenschaften von fehlertoleranten Systemen. Die Hauptprobleme, die festgestellt wurden, sind:

- Skalierbarkeit: Die Verarbeitung einer großen Anzahl an Zuständen in einem statischen Modellierungsverfahren ist problematisch. Es ist unrealistisch anzunehmen, dass industriell eingesetzte Hardware- und Software-Systeme mit klassischen analytischen Modellierungsansätzen skalierbar sind, bei denen jedes Systemelement einzeln beschrieben

werden muss.

- Steady-State-Metriken: Sämtliche untersuchten Werkzeuge bieten ausschließlich Steady-State-Metriken an.
- Benötigtes Systemwissen: Sämtliche untersuchten Werkzeuge, mit Ausnahme von FIGARO, erfordern umfangreiches Wissen über die eingesetzten Modellierungsmethoden, wie z.B. Markov-Ketten, Petri-Netze und endliche Automaten.
- Modellintegration: Mit Ausnahme von Möbius, ist es nicht möglich, verschiedene Einsatzbereiche mit unterschiedlichen Modellierungsverfahren abzubilden und diese Modelle dann zu integrieren.
- Modellierungsebene: Sämtliche untersuchten Werkzeuge sind technisch- und IT-orientiert. Kein Werkzeug ist für die Modellierung von Geschäftsprozessen einsetzbar.
- Sämtliche untersuchten Werkzeuge wurden primär für die Systementwicklung entworfen. Das Ziel ist dabei, zukünftiges Systemverhalten abschätzen zu können. Keines der untersuchten Werkzeuge ist zur automatischen bzw. semi-automatischen Modellierung eines bestehenden Systems einsetzbar.

2.6.1.2 Benchmarking-Werkzeuge

Benchmarking-Werkzeuge verwenden systemabhängige Verfügbarkeitsanalysen, die auf einem empirischen Test basieren. Ziel dieses Teils der Untersuchung war es, diejenigen Werkzeuge zu identifizieren, die quantifizierbare Verfügbarkeitsmessungen unterstützen. Diesbezüglich relevante Werkzeuge sind Eclipse Test und Performance Tool Plattform (TPTP), ExaustiF, FAIL-FCI, Networked Fault Tolerance und Performance Evaluator (NFTAPE) sowie QUAKE.

Eclipse TPTP ist bei weitem das komplexeste Werkzeug dieser Gruppe. Allerdings ist es nicht explizit auf die Verfügbarkeitsbestimmung und -messung ausgerichtet. Vielmehr stellt es ein generisches Framework zur Verfügung.

Andere wichtige Werkzeuge dieser Gruppe beruhen auf Fehlerinjektion (*Failure Injection*) und der Messung ihrer Auswirkungen. Diese Vorgehensweise unterstützt die empirische Verfügbarkeits-bestimmung des Zielsystems unter kontrollierten Fehlerbedingungen.

ExaustiF basiert auf Fehlerinjektion, bei der Ausfälle während verschiedener Entwicklungsphasen in das System eingegeben werden. Dieser Ansatz ist derzeit unter kontrollierten Bedingungen für

Echtzeitbetriebssysteme für Mehrprozessorsysteme durchführbar wie RTEMS / ERC32 und RTEMS / Intel.

FAIL-FCI wird zur Einspeisung von Ausfällen in Gridsysteme verwendet. Ausfälle werden mit einer FAIL-Sprache beschrieben, die dann in C++-Code umgewandelt wird. Dieser wird an verschiedenen Gridknoten kompiliert und ausgeführt. Auf diese Weise können unterrschiedliche Unterbrechungen simuliert und ihre Auswirkungen auf die Gridverfügbarkeit gemessen werden.

NFTAPE verfolgt ein ähnliches Konzept wie FAIL-FCI, ist in seiner Anwendung allerdings auf Kommunikationsnetze beschränkt.

Bei QUAKE werden Ausfälle in Webservice-Systeme eingespeist und ermöglichen online Messungen hinsichtlich Unterbrechung, Belastung und Systemintegrität.

Die Ergebnisse zur Untersuchung von Benchmarking-Werkzeugen für Verfügbarkeitsbestimmung lassen sich wie folgt zusammenfassen:

- Die untersuchten Werkzeuge sind entweder Mehrzweckwerkzeuge (z.B. Eclipse TPTP Platform) oder in Ihrer Anwendung auf eng begrenzte Nischen festgelegt (z.B. auf Kommunikationsnetzwerke).
- Nach unserem besten Wissen, sind die untersuchten Werkzeuge dieser Gruppe die einzigen Werkzeuge, die zur online Verfügbarkeitsermittlung eingesetzt werden können. Die Werkzeuge basieren jedoch alle auf Verfahren zur Fehlerinjektion. Hierzu müssen Fehlerbeschreibungen vor dem eigentlichen Test definiert werden. Im Gegensatz zu empirischen Black-Box-Methoden, erlauben diese Verfahren nicht die automatisierte Auswahl von wesentlichen Systemvariablen.

Qualitative Werkzeuge

Qualitative Werkzeuge verwenden zur Bestimmung der Systemverfügbarkeit beschreibende, tabellarische und diskrete Evaluierungstechniken, wie beispielsweise Audits, Protokolldateien, Interviews, Fragebögen und Implementierungsvorlagen. Die untersuchten Werkzeuge können in zwei Untergruppen geteilt werden:

2.6.1.3 Werkzeuge zur Risikoverwaltung

Werkzeuge aus dieser Kategorie approximieren Risiken auf der Prozess-Ebene. Von den untersuchten Werkzeugen sind für die vorliegende Studie folgende relevant: CASIS, Secura 17799, COBRA, CounterMeasures, CRAMM, EBIOS, GS-Tool, ISAMM, OCTAVE, PILAR, PROTEUS, RA2 und RiskWatch. Die überwiegende Zahl dieser Werkzeuge beruhen auf internationalen Standards wie ISO-27002 (früher ISO-17799), ISO 27001, US-NIST-800-26-Standards und COBIT-4.0. Die Werkzeuge verwenden mehrstufige Prozessmodelle, bei denen Kontext-relevante Informationen und Sicherheits-gefährdungen modelliert sowie Gefährdungsanalysen durchgeführt werden.

Die Ergebnisse zur Untersuchung von Werkzeugen zur Risikoverwaltung lassen sich wie folgt zusammenfassen:

- Verfügbarkeit wird nicht explizit als Risiko benannt.
- Alle Werkzeuge sind ausschließlich qualitativ und basieren auf Daten, die durch informelle Methoden erhoben werden. Die untersuchten Werkzeuge verwenden keine Datenüberwachungssysteme (*Monitoring*) und modellieren das laufende System nicht explizit.
- Sämtliche Empfehlungen zur Risikominimierung bzw. Verfügbarkeitsverbesserung beziehen sich auf die Planung und Strukturierung des jeweiligen Systems.

2.6.1.4 Werkzeuge zur Prozessverwaltung

Werkzeuge zur Prozessverwaltung sind selbst umfangreiche und komplexe Softwaresysteme, die zur Steuerung, Wartung und Optimierung von Hardware- und Software-Infrastrukturen verwendet werden. Aus methodischer Sicht werden sowohl formale als auch informelle Methoden eingesetzt, z.B. Workflow-Modellierung sowie Metadaten-, Gefährdungs- und Störungsanalysen. Die Mehrzahl der untersuchten Werkzeuge sind statisch. Eine Ausnahme bildet das Werkzeug HP-Mercury. Es ermöglicht Verfügbarkeitsbeobachtung zur Laufzeit und wird daher auch als Online-Benchmarking-Werkzeug klassifiziert. Weitere Werkzeuge dieser Gruppe sind: Software AG CentraSite, IBM High Availability Services, IBM Tivoli Availability Process Manager, Fujitsu Interstage Process Manager und HP Mercury. Bei den letzten drei Werkzeugen handelt es sich um Umgebungen mit dem Ziel der vollständigen Systemverwaltung auf Geschäftsprozessebene hinsichtlich Steuerung (*Governance*) und Optimierung. Diese Werkzeuge decken den Prozess-

Lebenszyklus über folgende Teilsegmente ab: Modellierung, Integration, Automatisierung, Entwicklung, Verwaltung, Wartung und Optimierung. Sämtliche untersuchten Werkzeuge verfügen über Abbildungsmechanismen zwischen Geschäftsprozess- und IT-Ebene. Hierzu werden Verfügbarkeits-, Risiko- und Sicherheitsparameter auf Geschäftsprozessebene analysiert. Die untersuchten Werkzeuge unterscheiden sich in der Art und Weise, wie Geschäftsprozesse beschrieben werden. Interstage verwendet formale Beschreibungs-methoden, die aus der Geschäftsprozessmodellierung bekannt sind, z.B. die Business Process Modelling Notation. IBM Tivoli bietet die Component Failure Impact Analysis (CFIA) sowie eine Verfügbarkeits- und Vorfallsverwaltung an, wie sie mittels ITIL spezifiziert wird. HP Mercury integriert Speichermodelle (*Configuration Management DataBase*) mit ITIL-Vorgaben, Workflow-Management und der ITIL-Enactment-Engine. IBM Tivoli ist nur in einer Suite von Werkzeugen zur Prozessverwaltung erhältlich. Beide Werkzeuge weisen daher ähnliche Funktionen auf und sind vergleichbar komplex. Gleiches gilt für Mercury, das erst kürzlich von HP erworben wurde.

Es ist wichtig festzuhalten, dass zum Einsatz eines der besprochenen Werkzeuge jeweils eine komplexe, unterstützende Infrastruktur installiert werden muss. Üblicherweise müssen aus Kompatibilitätsgründen Produkte des gleichen Unternehmens kombiniert werden.

CentraSite ragt aus den zuvor besprochenen Werkzeugen heraus. Es ist ein freies Framework zur Entwicklung von SOA-Governance-Lösungen und benötigt keine weitere Infrastruktur, ausgenommen von CentraSite-konformen Verzeichnissen. Durch die herstellerunabhängige Initiative will man Herstellerabhängigkeiten vermeiden. CentraSite basiert auf einem offenen, frei zugänglichen Webservice-basierten Dienst, der eine Infrastruktur zur Beschreibung von Prozessen und Services bietet. CentraSite bietet weiterhin transparente Definitionen von Verfahrensweisen (*Policys*), Änderungssteuerung, Verlässlichkeitsanalysen, Beobachtung und Reportgenerierung sowohl auf der Prozess- als auch auf der IT-Ebene. Auf dem Markt befinden sich mehrere Werkzeuge, die die CentraSite-Architektur als Grundlage nutzen, um eigene Dienste zur Prozessoptimierung und Prozessverfügbarkeit anzubieten, z.B. Interstage Process Manager.

Die Ergebnisse zur Untersuchung von Werkzeugen zur Prozessverwaltung lassen sich wie folgt zusammenfassen:

- Werkzeuge zur Prozessverwaltung sind komplexe, hochintegrierte Software-Werkzeuge. Ihr Einsatz erfordert eine genaue Planung sowie Investitionen in das Verwaltungs-Knowhow

und in die Infrastruktur. Eine Ausnahme hiervon bildet das auf offenen Standards beruhende CentraSite.

- Sämtliche untersuchten Werkzeuge sind in der Lage, Geschäftsprozess- und IT-Ebene aufeinander abzubilden. Die Auswertung, der Export und die Wieder- bzw. Weiterverarbeitung wird jedoch dadurch erschwert, dass keine freien Standards, wie z.B. SOAP/WSDL eingesetzt werden, sondern proprietäre Protokolle und Standards. CentraSite bildet durch den Einsatz offener Standards hiervon eine Ausnahme.
- Die in den untersuchten Werkzeugen verwendeten qualitativen Methodiken und Prozesse sind typischerweise ITIL-konform, obwohl es nicht immer eindeutig dokumentiert wurde.
- Die Grundidee der Prozessverwaltung ist die Definition von so genannten *Key Performance Indicators* (KPI). Verfügbarkeit kann als ein KPI definiert werden, wird jedoch nicht explizit unterstützt.
- Prozessmodelle sind segmentiert, wie z.B. Tivoli Unified Process (TUP).

Hybride Werkzeuge

Hybride Werkzeuge vereinen in sich quantitative und qualitative Methodiken zur Verfügbarkeitsbestimmung. Die relevanten Vertreter dieser Kategorie sind: Relex Reliability Studio, Reliability Center PROACT und LEAP, Reliasoft und PENPET.

Relex Reliability Studio erhebt den Anspruch, Markov-Modelle und Zuverlässigkeitsblockdiagramme (quantitativ) mit Risiko- und Kritikalitätsanalysen (qualitativ) zu vereinen. Jedoch lassen sich jeweils nur quantitative Methoden, wie beispielsweise Markov-Ketten und Zuverlässigkeitsblockdiagramme, oder qualitative Methoden, wie beispielsweise eine Kritikalitätsmatrix und eine Risikobewertungsanalyse, untereinander verknüpfen. Die Verbindung von quantitativen und qualitativen Methodiken ist nicht möglich. So kann z.B. ein Markov-Modell nicht mit einem Workflow-Modell kombiniert werden.

Reliability Center PROACT und LEAP sind zwei gesonderte Werkzeuge, die zusammen in einem Paket erhältlich sind. PROACT führt eine quantitative Root-Cause-Analyse durch. LEAP verwendet qualitative Methoden, wie Chancenanalysen (o*pportunity analysi*s) und FMEAs. Dieses Werkzeug ermöglicht die Verbindung zwischen quantitativen und qualitativen Modellen, so dass eine Rangliste der unerwünschten Ereignisse als Ergebnis für den mit PROACT-beschriebenen Vorgang

erstellt werden kann.

Reliasoft erhebt ebenfalls den Anspruch, beide Methodentypen zu integrieren. Frei wählbare Beschreibungen, die aus den Methoden FMEA bzw. FMECA stammen, lassen sich manuell in ein Flussdiagramm überführen, das unter Verwendung von Logikgattern simuliert werden kann.

PENPET bietet einen neuartigen, aus dem Chemiesektor übernommenen Modellierungsansatz und wird primär für Mehrprozessoranalysen eingesetzt. PENPET beschreibt Eigenschaften höherer Ebenen (*high-level*) qualitativ und Eigenschaften niedrigerer Ebenen (*low-level*) quantitativ. Für High-Level-Beschreibungen werden Strukturformeln, sog. Makromoleküle eingesetzt. Eigenschaften niedrigerer Ebenen, vergleichbar mit Molekülen oder Atomen, werden mittels allgemeiner stochastischer Petri-Netze dargestellt. Für Moleküle und Atome werden Ausfallraten detailliert modelliert. Die Makromoleküle werden qualitativ als zusammenhängende Atom- oder Molekül-Verbindungen modelliert. Zur Analyse werden Petri-Netze in Markov-Ketten automatisiert umgewandelt. Das Werkzeug bietet einen interessanten Ansatz zur Integration qualitativer und quantitativer Methoden bzw. Eigenschaften.

Die Ergebnisse zur Untersuchung von hybriden Werkzeugen lassen sich wie folgt zusammenfassen:

- Die Mehrzahl der untersuchten Werkzeuge erhebt den Anspruch, sowohl qualitative als auch quantitative Methoden zu unterstützen. Tatsächlich werden jedoch, mit Ausnahmen von PENPET, nur zwei Werkzeuge in einem Paket bereitgestellt.
- Unterstützung für qualitative Verfügbarkeitsbestimmung ist implizit und basiert auf der Failure-Impact-Analyse sowie Risikofaktor- und Kritikalitätsanalyse. Hier ist kein Unterschied zu rein qualitativen Werkzeugen festzustellen.

Kompatibilität und Nutzbarkeit

Um sämtliche oder zumindest einen Großteil der Besonderheiten eines Systems abbilden zu können, müssten nach aktuellem Wissensstand mehrere Werkzeuge kombiniert werden. Hierzu ist es wichtig, die Kompatibilität der untersuchten Werkzeuge hinsichtlich Austausch von Modellen oder Resultaten zu analysieren.

Zunächst ist festzuhalten, dass kein anerkannter Standard für den Austausch von Modellen und

Ergebnissen existiert. Dies limitiert die Möglichkeiten eines freien Datenaustausches erheblich. Die untersuchten Werkzeuge sind zu einem gewissen Grad mit gängiger Büroverwaltungssoftware kompatibel (z.B. Microsoft Office). Nur eine begrenzte Anzahl von Werkzeugen unterstützen den Import von Microsoft Visio-Diagrammen. Keins der untersuchten Werkzeuge unterstützt den Import oder Export von UML-Modellen.

Bei neueren kommerziellen als auch akademischen Werkzeugen ist die Handhabbarkeit gut. Sämtliche Werkzeuge bieten eine umfangreiche Benutzeroberfläche, die eine schnelle Modellierung und eine *Wizard*-basierte Bearbeitung der Modelleigenschaften ermöglicht. Viele der Werkzeuge besitzen umfangreiche Report-Generatoren, mit denen sich Ergebnisse in Textform, tabellarisch und graphisch erzeugen lassen.

Zusammenfassung

Die Ergebnisse der Analyse von Werkzeugen zur Verfügbarkeitsermittlung werden im folgenden Abschnitt zusammengefasst:

1. *Integration quantitativer und qualitativer Methoden*
   Mehrere hybride Werkzeuge nehmen für sich in Anspruch, quantitative und qualitative Methoden zu integrieren. Mit einer Ausnahme handelt es sich dabei um Werkzeuge, die zwar beide Methoden in einem Paket, jedoch keine funktionale Verknüpfung beider Methoden anbieten. Das Werkzeug PENPET bietet als einzige Ausnahme, eine integrierte Modellierungsvariante, die näher untersucht werden sollte. Im größeren Rahmen bleibt das Integrationsdefizit weiterhin bestehen.

2. *Verbindung von IT- und Geschäftsprozess-/ Service-Ebene*
   Diese Kernthematik ist essentiell bei der Spezifikation, Implementierung und im Management komplexer IT-Systeme. Durch die Lösung dieser Problematik würden mehrere wichtige Fragestellungen gelöst werden. Hierzu zählen z.B. die Verknüpfung von Verfügbarkeits-Spezifikationen auf Geschäftsprozess-Ebene mit der IT-Infrastruktur oder die Optimierung der Investitionsstrategie hinsichtlich Geschäftsfeldoptimierung. Nahezu alle Werkzeuge innerhalb der Gruppe „Prozessverwaltung“ nehmen für sich in Anspruch, diese Problematik gelöst zu haben. Bei realitätsnaher Betrachtung muss jedoch festgestellt

werden, dass zur Erreichung dieses Ziels a) erhebliche, zusätzliche Investitionen in die Soft- und Hardware des gleichen Herstellers getätigt werden müssen. Dadurch wird b) im Idealfall ein statisches Abbilden zwischen den beiden Ebenen ermöglicht. Eine dynamische Online-Abbildung, wie es in einem industrienahen, durch hohe Dynamik gekennzeichneten Umfeld nötig wäre, ist mit keinem der untersuchten Werkzeuge möglich. Eine Begründung hierfür ist in der *Prozess-Enactment-Engine* zu suchen, welche bislang nicht in der Lage ist, Abbildungen dynamisch zu erstellen. Das Werkzeug HP Mercury bildet in der Netzwerk-Nische eine Ausnahme. Dieses Werkzeug konzentriert sich auf die Beobachtung von Netzwerken und kann Ereignisse zu Geschäftsprozessen zuordnen. Eine weitere Ausnahme bildet das Werkzeug CentraSite, welches ein offenes Framework zur Beschreibung von Prozessen bietet. Aktuell unterstützt CentraSite allerdings nur Web-Services.

3. *Skalierbarkeit und Komplexität*

   Sowohl analytische Werkzeuge als auch Werkzeuge zur Prozessverwaltung stützen sich auf statische Systembeschreibungen. Diese Herangehensweise erfordert einen hohen Grad an Detailwissens über das System (White-Box-Ansatz). Der Anwender muss Systemzustände und Verfügbarkeitseigenschaften bis ins kleinste Detail beschreiben. Durch die kontinuierlich steigende Komplexität moderner IT-Systeme wird dieser Ansatz zunehmend unrealisierbarer. Der Ansatz ist wenig skalierbar und zudem fehleranfällig. Neuere Ansätze, basierend auf Grey-Box- und Black-Box-Verfahren, sind Gegenstand aktueller Forschung.

4. *Dynamische Online-Ansätze*

   Lediglich innerhalb der Benchmarking-Gruppe finden sich Werkzeuge mit Online-Fähigkeiten, die unter Umständen zur Modellierung dynamischer Service-Verfügbarkeit genutzt werden könnten. Diese Werkzeuge sind jedoch auf die Auswertung von Testfällen und Testsystemen begrenzt. Werkzeuge, die Black-Box-orientierte Ansätze unterstützen, sind nach unserem besten Wissen nicht erhältlich. Hierzu eignen sich gegebenenfalls traditionelle statistische Werkzeugsammlungen (z.B. R, S-Plus), die hier jedoch nicht explizit untersucht wurden.

5. *Ausfallvorhersage*

   Mehrere Werkzeuge nehmen für sich in Anspruch, dass Systemausfälle prognostiziert werden können. Diese Werkzeuge gehören zur Kategorie Elektronik- oder

Mechatronikbauteile. Die Vorhersage von Ausfällen basiert dort auf empirischen Erhebungen, die in vordefinierten Bibliotheken abrufbar sind oder durch Spezifikation des Alterungsprozesses durch den Anwender getroffen werden. Im kommerziellen Bereich sind keine Werkzeuge bekannt geworden, die dynamische Ausfallvorhersagen für ein laufendes, dezentralisiertes IT-System ermöglichen. Im akademischen Bereich werden hierzu Black-Box-Ansätze evaluiert.

6. *Erforderliches Wissen*

   Sämtliche untersuchten Werkzeuge erfordern besondere Fachkenntnisse im Bereich fehlertoleranter Systeme und formaler Modellierungsmethoden. FIGARO ist eine nennenswerte Ausnahme, da dieses Werkzeug mit einer objektorientierten Beschreibungssprache arbeitet. Das Ergebnis wird für den Nutzer transparent in eine modellbasierte Darstellungsart transformiert.

7. *Systembeobachtung und Datenerfassung*

   Werkzeuge aus den Gruppen Prozessverwaltung und Benchmarking bieten ebenso Verfahren zur Systembeobachtung und Datenerfassung. Diese Verfahren sind generisch und nicht auf Verfügbarkeitsbestimmung abgestimmt. Voraussichtlich muss zur Datenerfassung aus Betriebssystemen und Applikationen auf die Gruppe kommerzieller und freier Werkzeuge zur Systembeobachtung (*Monitoring*) ausgewichen werden.

3 Ergänzender Glossar

In Ergänzung zum Kapitel „Definitionen“ werden im folgenden wesentliche Konzepte und Begriffe, die im Umfeld Hochverfügbarkeit und Verfügbarkeitsbestimmung eingesetzt werden, beschrieben. Die vorgestellten Konzepte und Begriffe orientieren sich an im internationalen wissenschaftlichen Umfeld etablierten Definitionen, die überwiegend in Englisch vorliegen. Zur Einschränkung von Mehrdeutigkeiten stellen wir daher neben den deutschsprachigen auch die englischsprachigen Definitionen zur Verfügung.

Deutsch	***Englisch***
Komplex Allgemein: Ein Ganzes, dessen Teile vielfältig miteinander verknüpft sind.	**Complex** *Etymology*: Late Latin complexus totality, from Latin, embrace, from complecti 1: a whole made up of complicated or interrelated parts (Merriam-Webster)
Komplexität Die *Komplexität* eines Softwareprogramms, auch *Zeitkomplexität* genannt, wird in $O(N)$ angegeben. Eine Komplexität von $O(N)$ resultiert in *N* Operationen bei *N* Eingabeparametern. Eine Komplexität von $O(N^3)$ resultiert in N^3 Operationen. Diese Definition darf nicht mit *cyclomatic complexity* eines Softwareprogramms verwechselt werden, wie sie von McCabe et al. (1989) definiert wurde. *Cyclomatic complexity* misst die Anzahl linear abhängiger Pfade durch ein Programm oder ein Modul und wird berech-net als die Anzahl von Entscheidungspunkten plus eins. Diese Vorgehensweise liefert eine Ordinalzahl, die	**Complexity** *Time complexity,* synonymously also called *complexity,* of $O(N)$ results in *N* operations given *N* input parameters. A $O(N^3)$ process results in order of N^3 operations given *N* inputs. This definition is not to confuse with that of *cyclomatic complexity* of a program (McCabe et al. (1989)) which measures the number of linearly-independent paths through a program module, calculated as the number of decision statements plus one. This measure provides a single ordinal number that can be compared to the cyclomatic complexity of other programs.

Deutsch	*Englisch*
leicht mit denen anderer Pro-gramme oder Module verglichen werden kann.	
Dynamisches System Ein dynamisches System besteht aus einem Zustandsraum und den Regeln, um von einem in den nächsten Zustand zu evolvieren. Ein dynamisches System wird bestimmt durch seinen aktuellen Zustand sowie den Regeln, die eine Zustandstransformation erlauben. Mathematisch gesehen besteht ein dynamisches System $(\mathbf{X},\Psi)$ aus einem Zustandsraum $\mathbf{X}$ und einer Abbildungsfunktion $\Psi : \mathbf{X} \to \mathbf{X}$ die Elemente aus $\mathbf{X}$ nach $\mathbf{X}$ transformiert.	**Dynamical System (also, dynamic system)** We define a *dynamical system* as a space of possible states and a rule to evolve these states from one into the other. Its functional capacities are causally founded. A dynamical system is completely determined by the knowledge of its current state and the laws that govern its evolution. Examples include living, intelligent and social systems but also physical, robotic, and computing systems. Mathematically, a *dynamical system* $(\mathbf{X},\Psi)$ consists of a metric and compact space $\mathbf{X}$ and a continuous mapping $\Psi : \mathbf{X} \to \mathbf{X}$ that maps elements of the space $\mathbf{X}$ to the space $\mathbf{X}$.
Fehlertolerante Computersysteme Ein System, welches die Fähigkeit besitzt, trotz Fehlern die korrekten Ein-/Ausgabefunktionen aufrecht zu erhalten.	**Fault-Tolerant Computer Systems** A system that has the capability to continue the correct execution of its programs and input/output functions in the presence of faults.
Große, komplexe Softwaresysteme Ein großes, komplexes Softwaresystem benennt eine Klasse von Systemen, die sich aus einer Vielzahl von Komponenten zusammensetzen, deren Interaktionen nicht exakt verstanden werden und für die kein analytisches Modell bekannt ist. Die Wartung eines solchen Systems ist mit erheblichen Kosten verbunden. Aufgrund der hohen Komplexität dieser Systeme können die	**Large, Complex Software System** A *large* or *complex software system* refers to the general class of software systems which have large number of components whose interactions are poorly understood and analytical models of the system do not exist. Its maintenance infers considerable human and economic cost to its operator. Its failure may put human lives at risk or generate significant negative economic impact. The dynamics of a

Deutsch	***Englisch***
Dynamik und funktionalen Zusammenhänge nur unzureichend mit analytischen Werkzeugen und Verfahren abgebildet werden.	complex software system cannot be predicted based on rigorous analytical models.
Nichtfunktionale Eigenschaften von Computersystemen Nichtfunktionale Systemeigenschaften sind Eigenschaften, die der Endnutzer während der Systemlaufzeit beobachtet. Diese sind z.B. Leistung, Verfügbarkeit, Fehlertoleranz oder Echtzeitverhalten.	**Nonfunctional Properties of Computing Systems** The nonfunctional properties of a computer system are those properties that do not describe or influence the principal task / functionality of the system, but can be observed by end users during the systems runtime. Examples include performance, availability, fault tolerance, composability, real-time behavior, security as well as safety.
Nicht-parametrisch, stark robust Häufig geht man bei Modellierungsverfahren davon aus, dass annähernd normal verteilte Daten verarbeitet werden. Modellierungs-techniken mit diesen Rahmenbedingungen nennt man *robust*. *Stark robuste* Verfahren arbeiten auch mit Datenverteilungen, die nicht normal verteilt sind. Solche Verfahren werden auch *nicht-parametrisch* genannt. Folgende Modellierungstechniken können unterschieden werden: ● *parametrisch*: eine spezifische funktionale Form der Datenverteilung wird angenommen ● *nicht-parametrisch*: die funktionale Form der Datenverteilung wird vollständig durch die vorliegenden	**Nonparametric, Strongly Robust** In general one would like to have a modeling technique which is not much affected by the distribution of the data. Techniques that handle distributions which deviate somewhat from normal are called *robust*. There also exist *strongly robust* methods which are frequently also labeled *nonparametric* methods which also work if the data distribution is far from normal. ● *parametric*: specific functional form of the density model is assumed ● *nonparametric*: functional form of the density is entirely determined by the data ● *semiparametric*: number of adaptive parameters can be increased in a specific way (a priori knowledge vs.

Deutsch	*Englisch*
Daten bestimmt ● *semi-parametrisch*: die Anzahl der Modellparameter kann adaptiert werden, um der Datenverteilung gerecht zu werden (a priori Wissen vs. vollständig datenbestimmt)	completely data determined)
Echtzeit-Computersystem Systeme, die Serviceergebnisse innerhalb einer spezifizierten Zeit liefern.	**Real-Time-Computer Systems** Systems that deliver service to a user within a specified deadline (physical time or duration).
Responsives Computersystem Ein fehlertolerantes Echtzeitsystem, welches einen Service innerhalb bestimmter Zeitgrenzen liefert (Malek (1993)).	**Responsive Computer System** Fault-tolerant real-time systems that deliver satisfactory service in a timely manner (Malek (1993)).
Stochastischer Prozess Ein stochastischer Prozess ist ein Modell, welches die Wahrscheinlichkeitsstruktur einer Sequenz von Beobachtungen über die Zeit beschreibt. Eine Zeitreihe, die lediglich über eine endliche Zeitperiode beobachtet wurde, ist ein Beispiel für einen stochastischen Prozess.	**Stochastic Process** A stochastic process is a model that describes the probability structure of a sequence of observations over time. A time series is a sample realization of a stochastic process that is observed only for a finite number of periods.
Überlebensfähigkeit Die Fähigkeit eines Systems unter abnormen Rahmenbedingungen die gewünschten Leis-tungen zu erbringen.	**Survivability** System's ability to operate under abnormal conditions including predetermined faults or component failures.
Process-Enactment Ein Process-Enactment-Modell beschreibt die Parameter und Kontrollmechanismen, die für den Übergang von einer Prozessaktivität in die nächste notwendig sind.	**Process Enactment** A process enactment model describes the parameters and the control mechanisms that are needed to control the transfer of process activies.
Wizard	**Wizard**

Ein Wizard (Assistent) bezeichnet eine Benutzeroberfläche, mit der ein Anwender z.B. bei einer Softwareinstallation durch meherere Dialoge geführt wird. Dabei ist die Reihenfolge der Dialoge festgelegt, so dass es für den Anwender bei komplexeren Aufgaben möglicherweise leichter ist, diese durch-zuführen.	A wizard is a user interface element where the user is led through a sequence of dialogs. Unlike most modern user interface paradigms, the user is forced to perform the task in a specific sequence. However, for complex or infrequently performed tasks where the user is unfamiliar with the steps involved, it may make it easier for them to perform the task.

A Literaturliste

ACARA-2 (2007): ACARA-2, 2007, http://www.openchannelfoundation.org/projects/ACARA_II

Andersen and Fagerhaug (2006): Andersen, B. and Fagerhaug, T.; *Root Cause Analysis: Simplified Tools and Techniques*; ASQ Quality Press; 2006

AttackTree+ (2007a): AttackTree+ Project Website, 2007, http://www.isograph-software.com/atpover.htm

AttackTree+ (2007b): AttackTree+ Technical Specification, 2007, http://www.isograph-software.com/_techspecs/attacktree%2BV1TS.pdf

Avizienis et al. (2003): Avizienis, A., Laprie, J.-C., Randell, B., and Landwehr, C.; *Basic concepts and taxonomy of dependable and secure computing* ; IEEE Transactions on Dependable and Secure Computing; --;2004

AvSim+ (2007a): AvSim+ Project Website, 2007, http://www.isograph-software.com/avsover.htm

AvSim+ (2007b): AvSim+ Technical Specification, 2007, http://www.isograph-software.com/_techspecs/avsim32techspec.pdf

Axis (2007): Axis Ra2 Project Website, 2007, http://www.aexis.de

Babcock et al. (1987): Babcock P. S.,. Bong F., and Gai E.; *On the Next Generation of Reliability Analysis Tools*; The Charles Stark Draper Laboratory, MA NASA ContractorReport 178389; 1987

Balakfushnan, Raghavendra (1990): Balakfushnan M. and Raghavendra C.S.; *On Reliability Modeling of Closed Fault-Tolerant Computer Systems*; EEE Transactions on Computers, Vol. 39, No. 4; 1990

Bavuso (1982): Bavuso S. J.; *Advanced Reliability Modeling of Fault-tolerant Computer-based Systems*; NASA, TM-84501; 1982

Bavuso (1984): Bavuso S. J.; *A User's View of CARE III*; Reliability and Maintainability Symposium; 1984

Bavuso et al. (1985): Bavuso S. J., Dugan J. B., Trivedi K. S., Rothmann E. M., and Smith W. E.; *Analysis of Typical Fault-tolerant Architectures Using HARP*; IEEE Transaction on Reliability, Vol. 36; 1985

Bavuso et al. (1994): Bavuso S. et al.; *HiRel: Hybrid Automated Reliability Predictor Tool System*; NASA TP 3452; 1994

Beilner et al. (1989): Beilner H., Mäter J., Weissenberg N.; *Towards a Performance Modelling Environment: News on Hit*; Modelling Techniques and Tools for Computer Performance Evaluation; 1989

Bensberg (2001): Bensberg, F.; *Web Log Mining als Instrument der Marketingforschung . Ein systemgestaltender Ansatz für internetbasierte Märkte*; Deutscher Universitäts-Verlag; 2001

Berson et al. (1987): Berson S., de Souza E., and Muntz R.; *An Object Oriented Methodology for the Specification of Markov Models*; UCLA Technical Report CSD- 870030; 1987

Bishop (1995): Bishop C. M.; *Neural Networks for Pattern Recognition*; Clarendon Press, London; 1995

Bisson, Saint-German (2007): Bisson J. and Saint-German R.; *The BS 7799 / ISO 17799 Standard For a better approach to information security*; Callio White Paper; 2007

Bitterli (2007): Bitterli, P. R.; *Praxishandbuch COBIT. IT-Prozesse steuern, bewerten und verbessern.*; Dpunkt Verlag; 2007

Bobbio et al. (1999): Bobbio, Garg, Gribaudo, Horvath, Sereno, Telek; *Modeling Software Systems with rejuvenation, restoration and checkpointing through fluid petri nets*; Proc. 8th Int. Workshop on Petri Net and Performance Models (PNPM'99), Zaragoza, Spain, pp. 82-91; 1999

Booth (1967): Booth, T. L.; *Sequential Machines and Automata Theory*; John Wiley and Sons, Inc.; 1967

Bouissou (2002): Bouissou M.; *Boolean Logic Driven Markov Processes: A Powerful New Formalism forSpecifying and Solving very Large Markov Models*; PSAM6, Puerto Rico; 2002

Bouissou (2003): Bouissou, A new formalism that combines advantages of fault-trees and Markov models: Boolean logic Driven Markov Processes, 2002

Bouissou (2005): Bouissou M.; *Automated Dependability Analysis of Complex Systems with the*

KB3Workbench: the Experience of EDF R&D; The International Conference on ENERGY andENVIRONMENT, CIEM 2005, Romania; 2005

Bouissou et al. (1991): Bouissou M., Bouhadana H., Bannelier M., Villatte N.; *Knowledge modelling and reliabilityprocessing: presentation of the FIGARO language and associated tools*; Safecomp'91,Trondheim (Norway); 1991

Bouissou et al. (1991): Bouissou, M and Bouhadana, H and Bannelier, M. and Villatte, N.; *Knowledge modelling and reliability processing: presentation of the FIGARO language and associated tools*; Safecomp'91; --;1991

Bouissou et al. (2002): Bouissou M., Humbert S., Muffat S., Villatte N.; *Feedback on Knowledge Bases*; ESREL 2002, France; 2002

Bouissou et al. (2005): Bouissou M., Dutuit S., Maillard S.; *Reliability Analysis of a Dynamic Phased Mission System:Comparison of Two Approaches*; Modern Statistical and Mathematical Methods inReliability, Wilson A., Limnios N., Keller-McNulty S., Armijo Y. (eds), World Scientific,Singapore; 2005

Bouissou, Lefebvre (2002): Bouissou M., Lefebvre Y.; *A Path-Based Algorithm to Evaluate Asymptotic Unavailability forLarge Markov Models*; RAMS 2002, USA; 2002

Bouissou, Bon (2003): Bouissou M., Bon J.L.; *A new formalism that combines advantages of fault-trees and Markovmodels: Boolean logic Driven Markov Processes*; Reliability Engineering and System Safety,Band 82, Ausgabe 2; 2003

Bouissou, Muffat (2004): Bouissou M., Muffat S.; *High Level Representations for Markov Analysis of Complex Dynamic Systems*; IASTED Modeling and Simulation, USA; 2004

Box und Jenkins (1970): Box, G.E.P. and Jenkins, G.M.; *Time series analysis: Forecasting and control*; San Francisco: Holden-Day; 1970

BQR (2007): BQR Project Website, 2007, http://www.bqr.com

Bream (1995): Bream, B.; *Reliability Block Diagrams and Reliability Modeling*; NASA Lewis Research Center; --;1995

Breton et al. (2006): Breton E., Bouissou M., Aupied J.; *A New Tool for Reliability Studies of Electrical Networkswith Sand-by redundancies: OPALE*; PMAPS 2006, Stockholm; 2006

Broomhead and Lowe (1988): D. S. Broomhead and D. Lowe; *Multivariable functional interpolation and adaptive networks.*; Complex Systems, 2:321-355; 1988

BS/ISO-17799 (2005): British Standard Institute; *Information Technology Code of Practice for Information Security Management: BS ISO 17799*; BSI Standards; 2005

BSI (2006): BSI; *Leitfaden IT-Sicherheit. IT-Grundschutz kompakt.*; Bundesamt für Sicherheit in

der Informationstechnik; --;2006

Butler (1986a): Butler R.W.; *An Abstract Language for Specifying Markov Reliability Models*; IEEE Transactions on Reliability, vol 35; 1986

Butler (1986b): Butler R.W.; *The SURE reliability analysis program*; NASA TM 87593; 1986

Butler (2007): The Sure Program Homepage, 2007, http://shemesh.larc.nasa.gov/people/rwb/sure.html

Butler, Johnson (1995): Butler R. W. and Johnson S. C.; *Techniques for Modeling the Reliability of Fault-Tolerant Systems With the Markov State-Space Approach*; NASA Reference Publication 1348; 1995

Callio (2007): Callio Project Website, 2007, http://www.callio.com/

Cara-Faulttree (2007): Cara-Faulttree Project Website, 2007, http://www.sydvest.com/Products/Cara/prod_info.htm

Carer et al. (2003): Carer P., Bellvis J., Bouissou M., Domergue J., Pestourie J.; *A New Method for ReliabilityAssessment of Electrical Power Supply with Standby Redundancies*; PMAPS; 2003

CARMS (2007): CARMS Project Website, 2007, http://www.tc.umn.edu/~puk/carms.htm

Carrasco (2002): Carrasco J.A.; *Computationally Efficient and Numerically Stable Reliability Bounds for Repairable Fault-tolerant Systems*; IEEE Trans. on Computers, 51(3); 2002

Carrasco (2003a): Carrasco J.A.; *Computation of Bounds for Transient Measures of Large Rewarded Markov Models Using Regenerative Randomization*; 1.Computers and Operations Research,30(6); 2003

Carrasco (2003b): Carrasco J.A.; *Solving Dependability/performability Irreducible Markov Models Using Regenerative Randomization*; IEEE Trans. on Reliability, 52(3); 2003

Carrasco (2003c): Carrasco J.A.; *Transient Analysis of Rewarded Continuous-time Markov Models by Regenerative Randomization with Laplace-transform Inversion*; The Computer Journal, 46(1); 2003

Carrasco (2004a): Carrasco J.A.; *Solving Large Interval Availability Models Using a Model Transformation Approach2*; Computers and Operations Research, 31(5); 2004

Carrasco (2004b): Carrasco J.A.; *Transient Analysis of Some Rewarded Markov Models Using Randomization with Quasistationarity Detection*; IEEE Trans. on Computers, 53(9); 2004

Carrasco (2005a): Carrasco J.A.; *An Efficient and Numerically Stable Method for Computing Bounds for the Interval Availability Distribution*; Technical Report DMSD 2005 1, Departament d'Enginyeria Electr`onica, Universitat Polit`ecnica de Catalunya; 2005

Carrasco (2005b): Carrasco J.A.; *Transient Analysis of Large Markov Models With Absorbing*

States Using Regenerative Randomization; Communications in Statistics–Simulation and Computation,34(4); 2005

Carrasco (2006): Carrasco J.A.; *Two Methods to Compute Bounds for the Distribution of Cumulative Reward for Large Markov Models*; Performance Evaluation, 63(12); 2006

Carrasco, Sune (2007): METFAC User's GuidE, 2007, http://dit.upc.es/qine/tools/metfac/guide.pdf

Carrasco, Figueras (1986): Carrasco J.A., Figueras J.; *METFAC: Design and Implementation of a Software Tool for Modeling and Evaluation of Complex Fault-Tolerant Computing Systems*; Proceedings of FTCS 16, IEEE Computer Society Press, 424-429; 1986

Casis (2007): Casis Project Website, 2007, http://www.aprico-consult.com/

CASRE (2007): CASRE Project Website, 2007, http://www.openchannelfoundation.org/projects/CASRE_3.0

Castelli et al. (2001): V. Castelli, R.E. Harper, P. Heidelberger, S.W. Hunter, K.S. Trivedi, K. Vaidyanathan and W.P. Zeggert; *Proactive Management of Software Aging*; IBM JRD, Vol 45, No. 2, pp. 311-332; 2001

CentraSite (2007): CentraSite Project Website, 2007, http://www.softwareag.com/Corporate/products/centrasite/default.asp

Chen et al. (2002): Chen Mike , Emre Kiciman, Eugene Fratkin, Armando Fox and Eric Brewer; *Pinpoint: Problem Determination in Large, Dynamic Systems*; Proceedings of International Performance and Dependability Symposium, Washington, DC; 2002

Ciardo et al. (1989): Ciardo G., Muppala J.K., and Trivedi K.S.; *SPNP: Stochastic Petri Net Package*; Proceedings of 3rd International Workshop on Petri Nets and Performance Models; 1989

Clark and Pradhan (1995): Clark, J. A. and Pradhan, D. K.; *Fault Injection: A Method for Validating Computing-System Dependability*; Computer; --;1995

Clark et al. (2001): Clark G., Courtney T., Daly D., Deavours D., Derisavi S., Doyle J. M., Sanders W. H., and Webster, P.; *The Möbius Modeling Tool*; Proceedings of the 9th International Workshop on Petri Nets and Performance Models; 2001

Cobra (2007): Cobra Project Website, 2007, http://www.riskworld.net

Conway, Goyal (1986): Conway A. E. and Goyal A.; *Monte Carlo Simulation of Computer System Availability/reliability Models*; IBM Research Rep. RC 12459; 1986

CounterMeasures (2007): CounterMeasures Project Website, 2007, http://www.alionscience.com/index.cfm

Couvillion et al. (1991): Couvillion J., Freire R., Johnson R., Obal II W. D., Qureshi M. A., Rai M.,

Sanders W. H., and Tvedt J. E.; *Performability Modeling with UltraSAN*; IEEE Software; 1991

CPNTOOLS (2007): CPNTOOLS Project Website, 2007, http://wiki.daimi.au.dk/cpntools/cpntools.wiki

CRAMM (2007): CRAMM Project Website, 2007, http://www.cramm.com

Crane et al. (2003): Crane L. S., West S., and Andrews A.; *Analysis of OCTAVE Support for HIPAA Security/Privacy Standards*; ATI IPT Technical Report 03-; 2003

Crowell et al. (2002): Crowell Jonathan, Mark Shereshevsky, Bojan Cukic; *Using Fractal Analysis to Model Software Aging*; Lane Department of Computer Science and Electrical Engineering; 2002

Dohi et al. (2000a): Dohi T., K. Goseva-Popstojanova and K. S. Trivedi; *Analysis of Software Cost Models with Rejuvenation*; Proc. of the IEEE Intl. Symp. on High Assurance Systems Engineering, HASE-2000; 2000

Dohi et al. (2000b): Dohi T. , Goseva-Popstojanova Katerina, Trivedi Kishor S.; *Statistical Non-Parametric Algorithms to Estimate the Optimal Software Rejuvenation Schedule*; Dept. of Industrial and Systems Engineering, Hiroshima University, Japan and Dept. of Electrical and Computer Engineering, Duke University, USA; 2000

Dolny et al. (1983): Dolny L. J., Fleming R. E., and De Hoff R. L.; *Fault-tolerant Computer System DesignUsing GRAMP*; In Proceedings of the 1983 Annual Reliability andMaintainability Symposium; 1983

Dolny, Fleming (1983): Dolny L. J. and Fleming R. E.; *Evaluation of Reliability Models forFault-Tolerant Systems*; ORSA/TIMS, Orlando, FL; 1983

Dugan et al. (1985): Dugan J. B., Trivedi K. S., Geist R. and Nicola V.; *Extended Stochastic Petri Nets: Applications and Analysis*; Performance ‘84: Models of Computer System Performance; 1985

EBIOS (2007): EBIOS Project Website, 2007, http://www.ssi.gouv.fr/

ECLIPSE (2007): Eclipse Foundation Website, 2007, http://www.eclipse.org

ECLIPSE TPTP (2007): ECLIPSE TPTP Project Website, 2007, http://www.eclipse.org/TPTP

Exhaustif (2007): Exhaustif Project Website, 2007, http://www.exhaustif.es

FaultTree+ (2007a): FaultTree+ Project Website, 2007, http://www.isograph-software.com/ftpover.htm

FaultTree+ (2007b): FaultTree+ Technical Specification, 2007, http://www.isograph-software.com/_techspecs/psa32techspec.pdf

FIGARO (2007): FIGARO Project Website, 2007, http://www.edf.fr/72493m/txt/Home-fr/Research--Development/The-scientific Community/Downloads/KB3.html

Fleming (1980): Fleming R. E.; *Coherent System Repair Models*; Ph.D. dissertation, T.R.195, Dept. of Operations Research and Dept. of Statistics, Stanford University, USA; 1980

Fleming, Dolny (1984): Fleming R. E. and Dolny L. J.; *Fault-tolerant Design-to-specs with GRAMP & GRAMS*; In Proceedings of the 1984 Annual Reliability and MaintainabilitySymposium; 1984

FRACAS (1999): n/a; *Failure Reporting, Analysis and Corrective Action System (FRACAS) Application Guidelines*; SRC Conference ; --;1999

Frank (1990): Frank, P.M.; *Fault Diagnosis in Dynamic Systems Using Analytical and Knowledge-based Redundancy - A Survey and Some New Results*; Automatica, Vol. 26, No. 3, pp. 459-474; 1990

Fraser and Swinney (1986): Fraser A.M., Swinney H.L.; *Independent coordinates frostrange attractors from mutual inormation*; Physical review A33(2), pp.1134-1140; 1986

Freitas (1999): Joao F. G. de Freitas; *Bayesian Methods for Neural Networks*; Trinity College, University of Cambridge and Cambridge University Engineering Department, PhD Thesis; 1999

Gallois, Pillière (1991): Gallois M., Pillière M.; *Benefits Expected From Automatic Studies with KB3 in PSAs at EDF*; PSA99, Washington; 1991

Garg et al. (1995): Garg S., A. Puliafito M. Telek and K. S. Trivedi; *Analysis of Software Rejuvenation using Markov Regenerative Stochastic Petri Net*; In Proc. of the Sixth IEEE Intl. Symp. on Software Reliability Engineering, Toulouse, France; 1995

Garg et al. (1998a): Garg S., A. van Moorsel, K. Vaidyanathan and K. S. Trivedi; *A Methodology for Detection and Estimation of Software Aging*; In Proc. of the Ninth IEEE Intl. Symp. on Software Reliability Engineering, Paderborn, Germany; 1998

Garg et al. (1998b): Garg S., Puliofito A., Telek M., Trivedi K.; *Analysis of preventive Maintenance in Transaction based Software Systems*; Department of electrical & computer engineering Duke University Durham; 1998

Geist, Trivedi (1983): Geist R. & Trivedi K. S.; *Ultrahigh Reliability Prediction for Fault-tolerant Computer Systems*; IEEE Transactions on Computers, Vol. 12; 1983

Gill (1962): Gill, A.; *Introduction to the Theory of Finite-state Machines*; McGraw-Hill; 1962

Goyal et al. (1986a): Goyal A., Carter W. C., De Souza E. S, Lavenberg S. S., and Trivedi K. S.; *The System Availability Estimator*; Digest of the 16th Annual Symposium on Fault-Tolerant

Computing; 1986

Goyal et al. (1986b): Goyal A., Lavenberg S. S., and Trivedi K. S.; *Probabilistic Modeling of Computer System Availability*; IBM Research Rep. RC 11076; 1986

Goyal et al. (1987): Goyal A., Lavenberg S.S., and Trivedi K.S.; *Probabilistic Modeling of Computer System Availability*; Annals of Operations Research 8; 1987

GSTOOL (1992): BSI; *IT-Sicherheitshandbuch - Handbuch für die sichere Anwendung der Informationstechnik*; Bundesdruckerei; 1992

GSTOOL (2007a): GSTOOL – Das BSI Tool zum IT-Grundschutz, 2007, http://www.bsi.bund.de/gstool

GSTOOL (2007b): IT-Grundschutz-Kataloge, , http://www.bsi.bund.de/gshb

Harel (1987): Harel, D.; *Statecharts: A Visual Formalism for Complex Systems*; Sci. Computing Programming; --231-274;1987

Harel and Pnueli (1985): Harel, D. and Pnueli, A.; *On the Development of Reactive Systems*; Springer; 1985

HAS (2007): IBM High Availability Services, 2007, http://www-935.ibm.com/services/us/index.wss/offerfamily/bcrs/a1026936

Hastie and Tibshirani (1996): Hastie T., Tibshirani R.; *Discriminant adaptive nearest neighbor classification*; IEEE PAMI 18, pp. 607-616; 1996

Haverkort et al. (1992): Haverkort B. R., Niemegeers I. G., Veldhuyzen van Zanten P.; *DyQNtool – A Performability Modeling Tool Based on the Dynamic Queuing Network Concept*; Computer Performance Evaluation: „Modeling Techniques and Tools“, North-Holland; 1992

Haverkort, Niemegeers (1996): Haverkort B.R and Niemegeers I.G.; *Performability Modeling Tools and Techniques*; University of Twente, Tele-Informatics and Open Systems; 1996

Haverkort, Niemegeers (1996): Haverkort B.R. and Niemegeers I.G.; *Performability Modelling Tools and Techniques*; University of Twente, Tele-Informatics and Open System; 1996

Hertz et al. (1991): Hertz J., Krogh P., Palmer R.; *Introduction to the Theory of Neural Computation*; A Lecture Notes Volume in the Santa Fe Institute Studies of Complexity, Vol. I; 1991

Hines et al. (1999): Hines, Miller, Hajek; *A Hybrid Approach for Detecting and Isolating Faults in Nuclear Power Plant Interacting Systems*; Nuclear Technology MS #9188; 1999

Hirel et al. (2000): Hirel C., Sahner R.A., Zang X., Trivedi K.S.; *Reliability and Performability Modeling Using SHARPE 2000*; Proceedings of the 11th International Conference on Computer Performance Evaluation: Modelling Techniques and Tools; 2000

Hoare (1978): Hoare, C.A.R; *Communicating Sequential Processes*; Communication of the ACM; --;1978

Hoffmann (2005): Hoffmann G.; *Failure Prediction in Complex Computer Systems: A Probabilistic Approach*; Dissertation Humboldt-Universität Berlin, Shaker Verlag, ISBN 3-8322-4787-4 ; 2005

Hoffmann et al. (2006): Hoffmann, Trivedi, Malek ; *A Best Practice Guide to Resource Forecasting for the Apache Webserver*; 12th IEEE International Symposium Pacific Rim Dependable Computing (PRDC'06), University of California, Riverside, USA; 2006

Holzmann (1996): Holzmann G.J.; *Early Fault Detection Tools*; Bell Laboratories 2C-521, working paper; 1996

Hopkin and Moss (1976): Hopkin, D. and Moss, B.; *Automata*; New York: Elsevier North-Holland; 1976

Horau et al. (2005): Horau W., Tixeuil S., and Vauchelles, F.; *Easy Fault Injection and Stress Testing with FAIL-FCI*; LRI-CNRS 8623 et INRIA Grand Large; 2005

Howard (1971): Howard, R.; *Dynamic Probabilistic Systems*; John Wiley and Sons; 1971

Huang et al. (1995): Huang Y., C. Kintala, N. Kolettis and N. Fulton; *Software Rejuvenation: Analysis, Module and Applications*; In Proc. of the 25th IEEE Intl. Symp. on Fault Tolerant Computing (FTCS-25), Pasadena, CA; 1995

Humpert (2005): Humpert, F; *IT-Grundschutz umsetzen mit GSTOOL. Anleitungen und Praxistipps für den erfolgreichen Einsatz des BSI-Standards*; Carl-Hanser-Verlag München; 2005

IFIP WG 10.4 : INTERNATIONAL FEDERATION FOR INFORMATION PROCESSING; *WG 10.4 on DEPENDABLE COMPUTING AND FAULT TOLERANCE*; INTERNATIONAL FEDERATION FOR INFORMATION PROCESSING; 1980

Interstage (2007): Fujitsu Interstage Business Process Manager, 2007, http://www.fujitsu.com/global/services/software/interstage/bpm/index.html

ISAMM (2007): ISAMM Project Website, 2007, http://www.telindus.com

ITGI (2007): ITGI; *COBIT 4.0*; IT Governance Institute; --;2007

ITIL (2007): Continual Service Improvement (ITIL V3, CSI) , 2007,

itSMF (2007): itSMF Self-assesements Programs , 2007, http://www.itsmf.com/bestpractice/selfassessment.asp

ITU-T E.800: International Telecommunication Union; *QUALITY OF TELECOMMUNICATION SERVICES: CONCEPTS, MODELS, OBJECTIVES AND DEPENDABILITY PLANNING*; International Telecommunication Union; 2004-2007

Johnson and Malek (1988): Johnson, A. M. (JR) and Malek M.; *Survey of Software Tools for Evaluating Reliability, Availability, Serviceability*; ACM Computing Surveys; --227-269;1988

Kanoun, Blain (1996): Kanoun K. and Blain L.; *SURF-2 - A Tool for Modeling and Evaluation of Dependability Measures*; Proceedings of the 2nd International Computer Performance and Dependability Symposium; 1996

Kanoun, Borrel (2007): SURF-2 Homepage, 2007, http://www.laas.fr/surf/what-uk.html

Krafzig et al. 2004: Krafzig and K. Banke and D. Slama.; *Enterprise SOA: Service-OrientedArchitecture Best Practices*; Prentice Hall; 2004

Lala (1983a): Lala J. H.; *Mark1 Markov Modeling Package*; The Charles Stark Draper Laboratory; 1983

Lala (1983b): Lala J. H.; *Interactive Reductions in the Number of States in Markov Reliability Analysis*; Proceedings of the AZAA Guidance and Controls Conference; 1983

Lamport (1994): Lamport, L.; *The Temporal Logic of Actions"*; ACM Transactions on Programming Languages and Systems; --872-923;1994

Lapedes and Farber (1987): Lapedes A., Farber R.; *Nonlinear Signal Processing using Neural Networks; Prediction and System Modeling*; Technical Report: Los Alamos National Laboratory; 1987

Lepold (1991): Lepold R., PENPET: A Performability Modeling Evaluation Tool Based on Stochastic Petri Nets, 1991

Li et al. (2002): Li L., Vaidyanathan, Trivedi; *An Approach for Estimation of Software Aging in a Web Server*; Department of electrical & computer enegineering Duke University Durham; 2002

Littlewood and Strigini (2000): Bev Littlewood , Lorenzo Strigini,; *Software reliability and dependability: a roadmap*; Proceedings of the conference on The future of Software engineering, p.175-188, Limerick, Ireland; 2000

Lyu, Schoenwaelder (1998): Lyu M.R. and Schoenwaelder J.; *Web-CASRE: „A Web-Based Tool for Software Reliability Measurement"*; Proceedings of International Symposium on Software Reliability Engineering, Paderborn, Germany; 1998

Makam et al.(1982): Makam S., Avizienis A., and Grusas G.; *UCLA ARIES 82 users' guide*; Tech. Rep. CSD-82030, Computer Science Dept., Univ. of California, Los Angeles; 1982

Makam, Avizienis (1982): Makam S. V., and Avizienis A.; *ARIES 81: A reliability and life-cycle evaluation tool for fault-tolerant systems*; Digest of the 12th Annual Symposium on Fault-Tolerant Computing; 1982

Malek (1993): Malek, M.; *A Consensus-based Model for Responsive Computing*; IEICE Transactions on Information and Systems, Vol. E76-D, No. 11; 1993

Markov (1906): Markov, A. A.; *Rasprostranenie zakona bol'shih chisel na velichiny, zavisyaschie drug ot druga*; Izvestiya Fiziko-matematicheskogo obschestva pri Kazanskom universitete; --; 1906

Mauser (1990): Mauser M., Implementierung eines Optimierungsverfahrens für rekonfigurierbare Systeme, 1990

McCabe et al. (1989): McCabe, Thomas J. & Butler, Charles W.; *Design Complexity Measurement and Testing*; Communications of the ACM 32, 12, pp. 1415-1425; 1989

McDermott et al. (1996): McDermott, R. E. and Raymond, J. M. and Beauregard, M. R.; *The Basics of FMEA*; Productivity, Inc.; 1996

Meer, Sevcikova (1996): de Meer H. and Sevcikova H., Xpenelope User Guide, 1996

Meer, Sevcikova (1997): de Meer H. and Sevcikova H.; *PENELOPE: Dependability Evaluation and the Optimization of Performability*; Computer Performance Evaluation; 1997

Mercury (2007): Mercury BTO Enterprise Solutions, 2007, http://www.mercury.com/us/products/

METFAC (2007): METFAC Project Website, 2007, http://dit.upc.es/qine/tools/metfac/

Milner (1989): Milner, R.; *A Calculus of Communicating Systems*; Prentice-Hall; 1989

Möbius (2007): Möbius Project Website, 2007, http://www.mobius.uiuc.edu/

Molloy (1982): Molloy, M.K.; *Performance Analysis Using Stochastic Petri Nets*; IEEE Trans. on Computers; --913-917;1982

Moody and Darken (1989): Moody, Darken; *Fast learning in networks of locally tuned processing units*; Neural Computation 1:(2):281-294; 1989

Movaghar (1985): Movaghar, A., Performability Modelling with Stochastic Activity Networks, 1985

Movaghar, Meyer (1984): Movaghar A. and Meyer J.; *Performability Modeling with Stochastic Activiy Networks*; Proceedings of the Real-Time Systems Symposium; 1984

Mueller (1984): Mueller B.; *NUMAS: A Tool for the Numerical Modelling of Computer Systems*; Proc.International Conference on Modelling Techniques and Tools for Performance Analysis, Paris; 1984

Muppale and Lin (1996): Muppale, J. K. and Lin, C., Dependability of Large-Scale Distributed Software Systems Using Stochastic Petri Nets, 1996

Murata (1989): Murata, T.; *Petri Nets: Properties, Analysis and Applications*; Proceedings of the IEEE; --541-580;1989

NAP (2007a): Network Availability Program, 2007, http://www.isograph-software.com/napover.htm

NAP (2007b): Network Availability Program Technical Specification, 2007, http://www.isograph-software.com/_techspecs/nap32techspec.pdf

Neufelder (2000a): Neufelder A.M.; *How to measure the impact of specific development practices on fielded defect density*; Proceedings. 11th International Symposium on Software Reliability Engineering (ISSRE); 2000

Neufelder (2000b): Neufelder A.M.; *How to predict software defect density during proposal phase*; Proceedings of the IEEE National Aerospace and Electronics Conference (NAECON); 2000

Neville (1998): Neville S.W.; *Approaches for Early Fault Detection in Large Scale Engineering Plants*; PhD Thesis at University of Victoria, Canada; 1998

Ng, Avizienis (1977): Ng, Y. W., and Avizienis A.; *ARIES – an Automated Reliability Estimation System*; Proceedings of the 1977 Annual Reliability and Maintainability Symposium; 1977

Nowlan and Howard (1978): Nowlan, S. F. and Heap, H. F.; *Reliability Centered Maintenance*; Dep. of Defense; --;1978

Octave (2007): Octave Project Website, 2007, http://oattool.aticorp.org/

OpenSesame (2007): OpenSesame Project Website, 2007, http://www.lrr.in.tum.de/~walterm/OpenSesame/

Page et al. (1989): Page T.W., Berson S. E., Cheng W. C., Muntz , R. R.; *An Object Oriented Modeling Environment*; Proceedings of Conference on Object-oriented programming systems, languages and applications; 1989

Parker and Chua (1989): Parker T.S., Chua L.O.; *Practical Numerical Algorithms for Chaotic Systems*; Springer Verlag; 1989

Patton et al.(1989): Patton, R.J., P.M. Frank and R.N. Clark; *Fault Diagnosis in Dynamic Systems: Theory and Applications*; Prentice-Hall; 1989

Petri (1962): Petri, C.A., Kommunikation mit Automaten, 1962

Peyton Jones (1987): Jones Peyton, S. L. ; *The Implementation of Functional Programming Languages*; Prentice Hall; 1987

Pilar (2007): Pilar Project Website, 2007, https://www.ccn-cert.cni.es

Pnueli (1985): Pnueli, A.; *Applications of Temporal Logic to the Specification and Verification of Reactive Systems: A Survey of Current Threads"*; Lecture Notes in Computer Sciences; --; 1985

Poggio and Girosi (1990): Poggio T., Girosi F.; *A Theory of Networks for Approximation and*

Learning; Proc. of IEEE 78(9); 1990

Powell (1987): M. J. D. Powell; *Radial basis functions for multivariable interpolation: A review.*; J. C. Mason and M. G. Cox, editors, Algorithms for Approximation of Functions and Data, pp. 143-167; 1987

Proteus (2007): Proteus Project Website, 2007, http://www.infogov.co.uk

Puliafito et al. (1997): Puliafito A., Tomarchio O., and Vita L.; *Porting SHARPE on the WEB: Design and Implementation of a Network Computing Platform using Java Computer Performance Evaluation*; Proceedings of the 9th International Conference on Computer Performance Evaluation: Modelling Techniques and Tools; 1997

Reisig (1992): Reisig, W.; *A Primer in Petri Net Design*; Springer-Verlag; 1992

Relex (2007): Relex Project Website, 2007, http://www.relex.com

Reliability Center (2007): Reliability Center Project Website, 2007, http://www.reliability.com

Reliability Workbench (2007a): Reliability Workbench Project Website, 2007, http://www.isograph-software.com/_techspecs/avsim32techspec.pdf

Reliability Workbench (2007b): Reliabiity Workbench Technical Specification, 2007, http://www.isograph-software.com/_techspecs/wk32techspec.pdf

Reliasoft (2007): Reliasoft Project Website, 2007, http://www.reliasoft.com

Reliass (2007): Reliass Project Website, 2007, http://www.reliability-safety-software.de

RemedySuite (2007): RemedySuite Project Website, 2007, http://www.bmc.com/remedy/

Riggs (1982): Riggs, J. L.; *Engineering Economics*; McGraw-Hill, New York; 1982

RiskWatch (2007): RiskWatch Project Website, 2007, http://www.riskwatch.com

Rumelhart et al. (1986): Rumelhart, Hinton, Williams; *Learning internal representation by error propagation*; In Rumelhart, McCleland eds. Parallel Distributed Processing Volume I, MIT Press; 1986

Sabaton (2007): Sabaton Project Website, 2007, http://www.sydvest.com/Products/Sabaton/

Sahner at el. (2000): SHARPE 2000 Tool Manual, 2000, http://www.ee.duke.edu/~chirel/MANUAL/manualSharpe.pdf

Sahner, Trivedi (1986a): Sahner R. A. and Trivedi K. S; *A Hierarchical, Combinatorial-Markov Method of SolvingComplex Reliability Models*; Proceedings of the Full Joint Computer Conference AFIPS; 1986

Sahner, Trivedi (1986b): Sahner R. A. and Trivedi K. S.; *Reliability Modeling Using SHARPE*; IEEE Transactions on Reliability; 1986

Salfner (2005): Salfner, F.; *Predicting Failures with Hidden Markov Models*; Proceedings of the

fifth European Dependable Computing Conference (EDCC 5), Student Forum;Budapest; --; 2005

Salfner et al. (2004): Salfner F., Tschirpke S., Malek M.; *Comprehensive Logfiles for Autonomic Systems*; IEEE Proceedings of IPDPS 2004 (International Parallel and Distributed Processing Symposium); 2004

Salfner et al. (2004): Salfner, F., and Tschirpke, S., and Malek, M.; *Comprehensive Logfiles for Autonomic Systems*; Proceedings of 9th IEEE Workshop on Fault-Tolerant Parallel, Distributed and Network-Centric Systems, Santa Fe, New Mexico, USA; --;2004

Sanders (1995): Sanders W. H., Obal II W. D., Qureshi M. A., and Widjanarko F. K.; *The UltraSAN Modeling Environment*; Performance Evaluation; 1995

Sanders (2006): Möbius Manual, , http://www.perform.csl.uiuc.edu/mobius/manual/MobiusManual.pdf

Sanders, Meyer (1986): Sanders W. H. and Meyer J. F.; *METASAN: a Performability Evaluation Tool Based on Stochastic Activity Networks*; Proceedings of 1986 ACM Fall Joint Computer Conference; 1986

Schneeweiss (1999): Schneeweiss, W. G.; *Die Fehlerbaummethode*; LiLoLe-Verlag Hagen/Westfalen; 1999

Schoelkopf and Smola (2002): Schoelkopf B., Smola A.; *Learning with Kernels*; MIT Press; 2002

Sewera (2005): Sewera, S.; *Referenzmodelle im Rahmen von IT-Governance*; Wirschaftsuniversität Wien; 2005

Sharma, Bazovsky (1993): Sharma T.C. and Bazovsky I.; *Reliability Analysis of Large System By Markov Techniques*; Proceedings of IEEE Annual Reliability and Maintainability Symposium; 1993

Sharpe (2007): Sharpe 2002 Project Website, 2007, http://www.ee.duke.edu/~kst/software_packages.html

Shereshevsky et al. (2001): Shereshevsky Mark, Jonathan Crowell and Bojan Cukic; *Multifractal Description of Resource Exhaustion Data in Real Time Operating System*; Technical Report, West Virginia University; 2001

Shereshevsky et al. (2003): Shereshevsky Mark, Jonathan Crowell, Bojan Cukic, Vijai Gandikota, Yan Liu; *Software Aging and Multifractality of Memory Resources*; International Conference on Dependable Systems and Networks (DSN-2003), San Francisco; 2003

Shooman (1987): Shooman, M.; *Software Engineering: Design, Reliability and Management*; McGraw-Hill; 1987

Siewiorek and Swarz (1992): Siewiorek, Swarz; *Reliable Computer Systems Design and Evaluation*; The Digital Press, 2nd edition; 1992

SMERFS (2007): SMERFS Project Website, 2007, http://www.slingcode.com/smerfs/

Smith, Lala (1986): Smith T. B., and Lala J. H.; *Development and Evaluation of a Fault-tolerant Multiprocessor (FTMP) Computer*; FTMP executive summary NASA-CR-172286; 1986

Smyth (1994a): Smyth P.; *Hidden Markov Models for Fault Detection in Dynamic Systems*; Pattern Recognition, Vol. 27 No. 1; 1994

SoftRel (2007): SoftRel Project Website, 2007, http://www.softrel.com/

Some et al. (2001): Some, R. R. and Kim, W. S. and Khanoyan, G. and Callum, L. and Agrawal, A. and Beahan, J. J.; *A Software Injection Fault Methodology for Design and Validation of System Fault Tolerance*; International Conference on Dependable Systems; --;2001

SPNP (2007a): SPNP Project Website, 2007, http://www.ee.duke.edu/~kst/software_packages.html

SPNP (2007b): SPNP Users Manual, 2007, http://www.ee.duke.edu/~chirel/MANUAL/manual.pdf

Stalnaker (2007): ARCARA Users Manual, 2007, http://naca.larc.nasa.gov/search.jsp

Stamatis (1995): Stamatis, D. H.; *Failure Mode and Effect Analysis: FMEA from Theory to Execution*; American Society for Quality; --;1996

Stateflow (2007): Stateflow Project Website, 2007, http://www.mathworks.com

Stiffler et al. (1979): Stiffler J. J., Bryant L. A. and Guccione L.; *CARE III Final Report*; Contractor Rep. 159122 & 159123, NASA; 1979

Stonebumer et al. (2002): Stonebumer, A. and Goguen, A. and Fringa, A.; *Risk Management Guide for Inrormation Technology Systems*; NSI; --;2002

Storey (1996): Storey, N.; *Safety-Critical Computer Systems*; Addison-Wesley Longman, New York; 1996

Stott (2000): Stott D.T.; *Automated Fault Injection Based Dependability Analysis of Distributed Computer Systems*; Ph.D. Thesis, Univ. of Illinois; 2000

Stott et al. (2000): Stott D.T., Floering B., Kalbarczyk Z., and Iyer R.K.; *Dependability Assessment in Distributed Systems with Lightweight Fault Injectors in NFTAPE*; Proceedings of IPDS-4; 2000

Stott et al. (2002): Stott D.T., Jones P.H., Hamman M., Kalbarczyk Z. and Iyer R.K.; *NFTAPE: Networked Fault Tolerance and Performance Evaluator*; Proceedings of the International Conference on Dependable Systems and Networks; 2002

Tang and Iyer (1996): Tang, D. and Iyer, R. K.; *Experimental Analysis of Computer System Dependability*; Prentice-Hall; 1996

Tivoli (2007): IBM Tivoli Availability Process Manager, 2007, http://www-306.ibm.com/software/tivoli/products/availability-process-mgr/

Tixeuil et al. (2006): Tixeuil S., Hoarau W.. and Silva L.; *An Overview of Existing Tools for Fault-Injection and Dependability Benchmarking in Grids*; CoreGRID Technical Report; 2006

Trivedi (2002): Trivedi K.S.; *SHARPE 2002: Symbolic Hierarchical Automated Reliability and Performance Evaluator*; Proceedings of International Conference on Dependable Systems and Networks; 2002

Trivedi (2003): Trivedi K.S.; *Importance Analysis with Markov Chains*; ; --;2003

Trivedi (2007): SHARPE 2000 GUI Manual, 2007, http://www.ee.duke.edu/~chirel/MANUAL/gui.pdf

Ulerich and Powers (1988): Ulerich, N. H., & Powers, G. A.; *Online hazard aversion and fault diagnosis in chemical processes: the digraph/fault tree method*; IEEE Transactions on Reliability 37 (2), pp. 171-177.; 1988

Vaidyanathan and Trivedi (1999): Vaidyanathan K. and K. S. Trivedi; *A Measurement-Based Model for Estimation of Resource Exhaustion in Operational Software Systems*; In Proc. of the Tenth IEEE Intl. Symp. on Software Reliability Engineering, Boca Raton, Florida; 1999

Van Bon et al. (2006): Van Bon, J. and Van der Veen, A. and Pieper, M.; *Foundations in IT Service Management, basierend auf ITIL*; Van Haren Publishing; 2006

Venkatasubramanian et al. (2003): Venkatasubramanian V., Rengaswamy R., Yin K., Kavuri S.; *A review of process fault detection and diagnosis Part I-III*; Computers and Chemical Engineering 27, pp. 293-311; 2003

Walter et al. (2007): Walter M., Siegle M., and Bode A.; *OpenSESAME - The Simple but Extensive, Structured Availability Modeling Environment*; Reliability Engineering & System Safety; 2007

Weigend et al. (1994): Weigend A. S., Gershenfeld N. A., eds.; *Time Series Prediction*; Proceedings of the Santa Fe Institute, Vol. XV; 1994

Weiss (1999): Weiss G. M.; *Timeweaver: a Genetic Algorithm for Identifying Predictive Patterns in Sequences of Events*; Rutgers University and AT&T Labs; 1999

Weiss (2001): Weiss G. M.; *Predicting telecommunication equipment failures from sequences of network alarms*; In W. Kloesgen and J. Zytkow, editors, Handbook of Data Mining and Knowledge Discovery. Oxford University Press; 2001

Wells (2006): Wells, L.; *Performance analysis using CPN tools*; Proceedings of the 1st International Conference on Performance Evaluation Methodologies and Tools; 2006

West, Andrew (2003): West S. and Andrew A. D.; *OCTAVE-Best Practices Comparative Analysis*; ATI IPT Technical Report 03-4; 2003

Zadach (1998): Zadach M., Wege der transienten Optimierung – ein zeitdiskreter Ansatz für PENELOPE, 1998